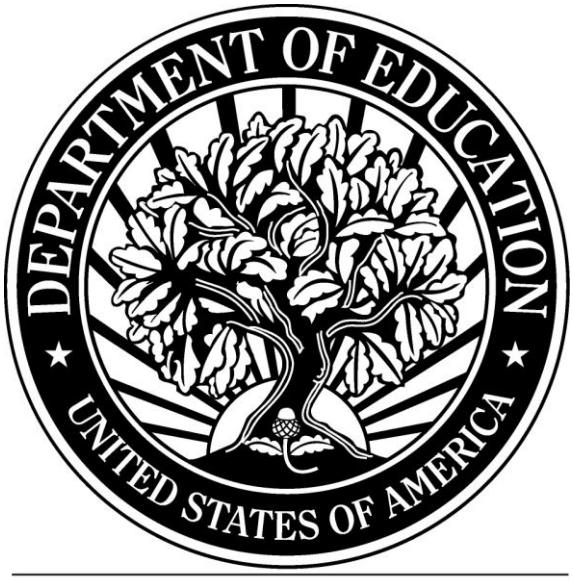
Student Aid Internet Gateway

Host Communication Guide for Mainframe & Midrange Users

Version 3.4



FEDERAL STUDENT AID

DOCUMENT CONTROL

DOCUMENT INFORMATION

Title:	SAIG Host Communication Guide
Revision:	3.4
Issue Date:	8/13/2012

DOCUMENT HISTORY

Version Number	Date	Summary of Change
3.0	1/12/2010	Updated for SAIG 3.2 release
3.1	1/19/2012	Updated Windows Installation section for SAIG 3.2 release
3.2	5/4/2012	Updated TDClient 3.2 Supported Platforms
3.3	5/23/2012	Updated TDClient 3.2 supported platforms for AIX
3.4	8/13/2012	Updated TDClient 3.2 supported platforms OS/400, HP-UNIX and Linux

Table of Contents

1 Overview

1.1 Preface

Welcome to the U.S. Department of Education's Federal Student Aid (FSA) Student Aid Internet Gateway (SAIG) that offers Title IV-eligible post-secondary institutions, third-party servicers, state agencies, lenders and guarantors, a secure, Internet-based method of exchanging Title IV data with the FSA Application Systems. The SAIG replaces what was formerly known as "TIV WAN" by moving Title IV transmissions from the General Electric (GEIS) value-added network to the Internet.

This guide is designed to meet the reference needs of programmers and data processing staff who transmit Title IV Data via a mainframe or midrange computer. Additionally, this guide serves as a working document that we will periodically update and revise so that you have access to the most current information possible.

The SAIG is designed around FSA's vision and target architecture to provide an Internet solution for data transmissions. FSA to the Internet offers an integrated solution for FSA's constituents by implementing a Commercial Off-the-Shelf (COTS) application that supports multiple hardware and operating system platforms.

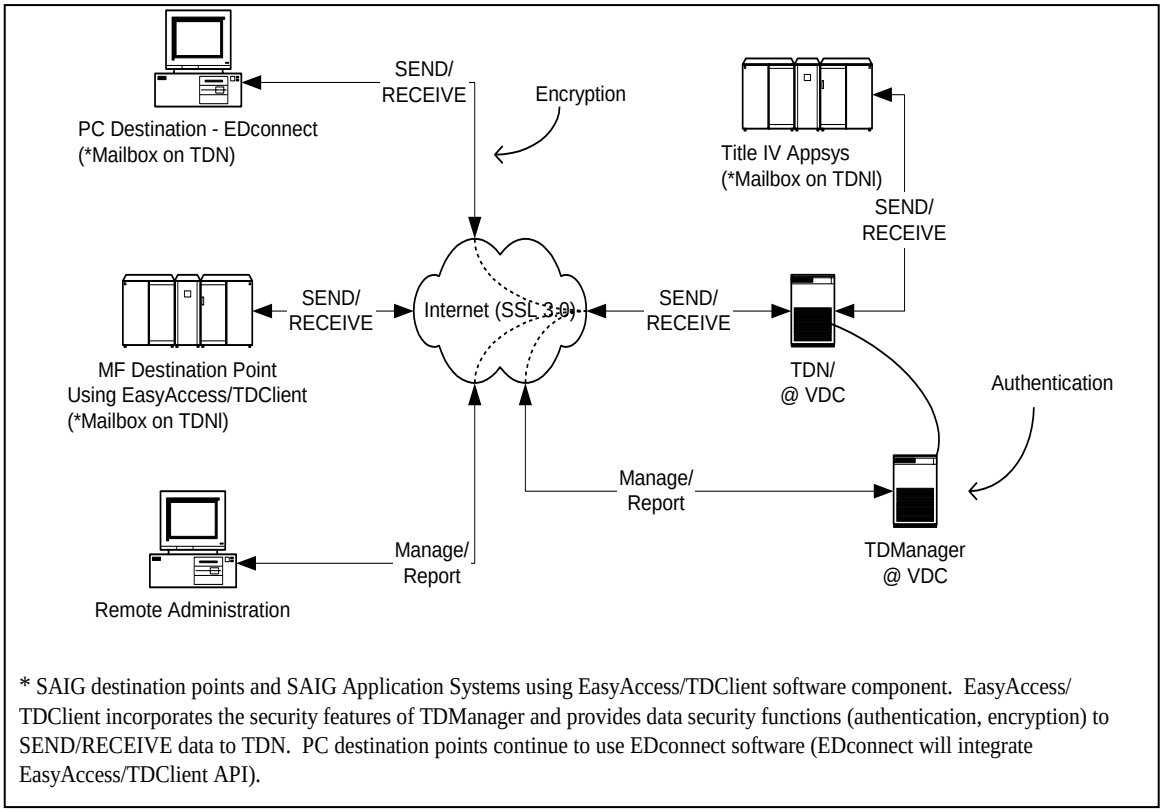
Note: To all third party software providers:

1. Do not include transmission headers and trailers (O*N) on files to be transmitted via EDconnect.

2. Use the appropriate technical reference when creating output. The application system receiving the data will dictate use of low values and null values.

3. Provide a Carriage Return/Line Feed (CR/LF) in the final position of the data file to be transmitted.

The diagram in **Figure 1** represents the flow of data between SAIG destination points and Application Systems.



* SAIG destination points and SAIG Application Systems using EasyAccess/TDClient software component. EasyAccess/ TDClient incorporates the security features of TDManager and provides data security functions (authentication, encryption) to SEND/RECEIVE data to TDN. PC destination points continue to use EDconnect software (EDconnect will integrate EasyAccess/TDClient API).

Figure 1: System Diagram

The integrated solution consists of TDClient, TDManager, TDNgine (TDN), and TDCommunity Manager components. The following sections provide more detail on each of these products.

1.1.1 TDClient

TDClient is the client software used to send and receive FTP (File Transfer Protocol) Title IV data transmissions securely over the Internet using FIPS 140-2 Advanced Encryption Standard (AES). Port 26581 needs to be open in your firewall to allow outbound TCP/IP traffic. You can request the TDClient software by contacting CPS/SAIG Technical Support at 800/330-5947 or by email at **CPSSAIG@ed.gov.**

The TDClient software has the compression and decompression steps built into it. This means you no longer need the separate steps in your MVS JCL or UNIX scripts for sends and receives.

TDClient is supported under the following mainframe/mid-range operating systems:

Supported Platforms for TDClient						
Windows	**Linux**	**Solaris**	**HP-UX PA-RISC**	**AIX**	**MVS (OS/390)**	**OS/400**
• Vista • Win7 • Server 2003 • Server 2008	• RedHat ES3x • RedHat ES4x • RedHat ES5x • RedHat ES6x • Suse ESx • Ubuntu • CentOS	• 9 Sparc • 10 Sparc • 11 Sparc	• 11i v2(11.23) • 11i v3(11.31)	• 6.1 • 7.1	• z/OS 1.10 • z/OS 1.11 • z/OS 1.12	• V5R4 • 6.1+

1.1.2 TDManager and TDCommunityManager

The Transaction Delivery Manager (TDManager) is the Click Commerce product that is used by System Administrators to manage SAIG. The TDCommunity Manager or TDCM is the Click Commerce product that is used to manage SAIG destination points. The product runs as a thin client and can be accessed via the Web. Users of the system are system administrators, customer service/technical support staff, and SAIG destination points. Destination points can use this system to manage their mailbox and view network traffic via the Internet.

The TDCM (formerly OSM) User's Guide, containing instructions on how to query your SAIG mailbox, is available on the FSAdownload Web site at fsadownload.ed.gov.

2 Installation Guide

2.1 Configuring and Installing TDClient

Student Aid Internet Gateway (SAIG) destination points and application systems can use TDClient in any of the following environments:

- MVS (OS/390), z/OS 1.10 + (must have POSIX functionality in LE, v1.9)
- OS/400, v5R4, 6.1+
- HP-UNIX, 11i v2(11.23) or 11i v3(11.31)
  Note: Only the PA-RISC processor is supported for HP-UX
- AIX, v6.1 or 7.1
- Sun Solaris, 9 + (SPARC chip only, Intel chip not supported)
- Linux (Redhat, SuSE, Ubuntu and CentOS)
- Windows XP, Windows 7, Windows Vista and Windows Server 2003 and 2008 Standard Edition

2.1.1 Helpful Hints

- TDClient requires a physical connection to the Internet.

  The TDClient software and the accompanying documentation are available for download at the FSA Download website https://www.fsadownload.ed.gov/softedconnect.htm.

2.2 Installation on MVS OS/390 Systems

To use TDClient, you must have MVS OS/390 z/OS 1.10 or above, with the MVS feature of Language Environment Version 1, Release 9 with POSIX functionality. Higher versions of MVS and OS/390 must include the appropriate C++ language support feature. In order to use the file transmission feature, you must also have installed and configured TCP/IP for MVS Version 3, Release 1 or higher.

2.2.1 Materials required prior to installation:

- Download the MVS z/OS TDClient from the FSA Download website at **https://www.fsadownload.ed.gov/softedconnect.htm.**
  - TDACCESS.V32.XMIT

  Download the TDClient.INI file from **https://www.fsadownload.ed.gov/softedconnect.htm**

2.2.2 To install TDClient:

1. FTP the TDACCESS.V32.XMIT file in BINARY mode to an MVS dataset with the following attributes: RECFM=FB, LRECL=80, and BLKSIZE=3120. The TDACCESS.V32.XMIT file contains the TDClient load library, example JCL and configuration files. You can FTP the file in a variety of ways, such as from a DOS ftp prompt (see Figure 2-1), ftp client software, or 3270 emulator.

```
C:\> ftp
ftp> open your.ip.address                                  <= connect to MVS/OS390
220 User (none)): userid                                   <= enter USERID
331 Enter password:xxxxxxx                                 <= enter PASSWORD
230 USERID logged on.
ftp> bin                                                   <= binary mode
200 Representation type is binary IMAGE.
ftp> quote site recfm=fb lrecl=80 blksize=3120             <= file attributes
200 SITE COMMAND WAS ACCEPTED
ftp> put c:\ TDACCESS.V32.XMIT 'your.xmit.dataset'         <=ftp the file to the
                                                           mainframe
200 PORT subcommand request successful
125 Storing data set user.ealib.file
250 Transfer completed successfully
ftp> quit                                                  <= disconnect
```

Figure 2: Example FTP from a DOS FTP Prompt

2. Upload the compressed file from step 4 into a Partitioned Data Set (PDS). To do this:

   a. Go to a **TSO READY** prompt.

   b. Type **RECEIVE INDA ('your.xmit.dataset')**. Replace "your.xmit.dataset" with the dataset you created in Step 4. See **Figure 2**.

   c. When prompted to "enter restore parameters", type **DA ('your.install.dataset')**. See **Figure 3**. Replace "your.install.dataset" with a dataset name appropriate for your installation. The install.dataset must be a different name than the dataset name used in Step 5b.

The steps above will create an Installation Library containing the files required to complete the installation of TDClient.

```
READY
RECEIVE INDA ('your.xmit.dataset')
Dataset SP01.DDNAME.INSTALL from SP)! On NODENAME
Enter restore parameters or 'DELETE" or END' +
DA ('your.install.dataset)
```

Figure 3: Example TSO RECEIVE

3. Edit the $INSTALL member of the **your.install.dataset** and make the changes described in Steps 1-6 of **Figure 4** below.

   Run the $INSTALL JCL.

```
 //jobname  JOB (acct),pgmr,MSGLEVEL=1,REGION=7M,CLASS=A,
 //  MSGCLASS=X,NOTIFY=user
 //*
 //* MEMBER $INSTALL
 //*
 //* TDClient/MVS Installation JCL.
 //*
 //* Make the following changes:
 //*
 //*  1) Provide the appropriate fields on the JOBCARD, above.
Change all occurrences of your.install.dataset to the name you created for this
 dataset.
Change all occurrences of your.user.tdload to a valid destination dataset name.
Change all occurrences of your.user.tdsamp to a valid destination dataset name.
Change all occurrences of your.user.tdssamp to a valid destination dataset name.
Change all occurrences of your.user.cpsamp to a valid destination dataset name.
 //*
 //*****************************************************************
 //*TSO Receive for DISTLIB and SAMPLIB Datasets.
 //*****************************************************************
 //*
 //RECEIVE  EXEC PGM=IKJEFT01,REGION=4096K
 //SYSTSPRT DD  SYSOUT=*
 //EALOAD   DD DSN=your.install.dataset(TDLOAD),DISP=SHR
 //EASAMP   DD DSN=your.install.dataset(TDSAMP),DISP=SHR
 //CPDBRM   DD DSN=your.install.dataset(TDSSAMP),DISP=SHR
 //CPSAMP   DD DSN=your.install.dataset(CPSAMP),DISP=SHR
 //SYSTSIN  DD  *
 RECEIVE INFILE(TDLOAD)
        DATASET('your.user.eaload')
 RECEIVE INFILE(TDSAMP)
        DATASET('your.user.easamp')
 RECEIVE INFILE(TDSSAMP)
        DATASET('your.user.cpdbrm')
 RECEIVE INFILE(CPSAMP)
        DATASET('your.user.cpsamp')
 /*
```

Figure 4: Example MVS OS/390 Install File before Editing

4. Allocate a new file with attributes LRECL=80, RECFM=FB, BLKSIZE=23440 and name the file **'your.dataset.prefix.TDCLIENT.EXFER.INI'**. This file will remain blank until you send your first file using TDClient, at which time it will populate with parameters contained in the TRANSFER command line of your JCL (described in *Section 4, Communication Procedures*).

5. Upload the **TDCLIENT.INI** file (from Step 1) as BINARY with the attributes RECFM=FB, LRECL=80, CRLF, and name the file **'your.dataset.prefix.TDCLIENT.INI'**.

Note: This file holds network configuration information and is described in *Section 4, Communication Procedures*. **Do not alter this file.**

2.3 Installation on HP/ SUN/AIX UNIX and LINUX Systems

TDClient provides file transfer capabilities with compression and encryption for UNIX and LINUX platforms.

2.3.1 Materials required prior to installation:

- Download the appropriate TDClient for your system from the FSA Download website https://www.fsadownload.ed.gov/softedconnect.htm
  - tdaccess_v32_FSA_HPUX11.SFX
  - tdaccess_v32_FSA_solaris.SFX
  - tdaccess_v32_FSA_AIX.SFX
  - tdaccess_v32_FSA_linux.SFX
- Download the new TDClient.INI file from **https://www.fsadownload.ed.gov/softedconnect.htm**

2.3.2 To install TDClient:

1. Make a directory called **TDC** on the UNIX or LINUX box you are using by typing **mkdir TDC** from the command prompt. You will choose the location to create the TDC directory.

   Note: To verify your folder or directory location, enter the command **pwd**.

2. Type **cd TDC** from the command prompt and press **Enter.**

3. FTP the file that you downloaded to the **TDC** folder.
   - Ensure that the TDClient install file has executed, read, and write permissions.

   Note: You can change the permissions with many GUI FTP programs or by using the chmod command at the command prompt by typing **chmod [+x +r +w]** tdaccess_v32_FSA_HPUX11.SFX.

4. Run the TDClient self-extracting file from the command line to expand its components by typing the appropriate file name and then press **Enter**.

   Note: You will see the file decompressing at this time.

5. You will be prompted with a default directory location of TDAccess3.2 to install TDClient.
   - The next prompt will say: “Directory does not exist. Create Directory?”
   - Type **“Y”** for yes and press the **enter** key

- Next you will be prompted to enter an e-mail address. You can enter any value here because the SAIG system does not use this e-mail address anywhere.
- If you are installing the UNIX/LINUX version of the TDClient you will be asked: "Install TDServer? <Y or N>". Enter "**N**" and press the **Enter** key. *Do not install TDServer.*

6. The program will begin installing and display this message:

   Installing TDAccess in directory /home/jtest/TDAccess3.2...

   Decompressing TDAccess 3.2 Installation file (this may take a minute or two ...)

   Note: you will see the files decompressing at this time.

7. After installing you will see the text:

   Creating TDAccess subdirectories...
   Resetting file permissions for TDAccess files...
   cp -p exfer.ini /home/jsteapp/TDAccess3.2/exfer.ini
   cp -p tpaddrss.ini /home/jsteapp/TDAccess3.2/tpaddrss.ini
   Installation completed!

8. After installation is completed you will see the following text:

   IMPORTANT SYSTEM CONFIGURATION INFORMATION:

   You must update your SHLIB_PATH environment variable to include the directory into which the TDAccess product was installed.*

   An example of how to do this is given in the file .profile_example created in this directory.

   Since you can control the value of SHLIB_PATH from your .profile, or from some other point, this install program will not attempt to make this change for you.

   However, the TDClient program will not be able to find the required shared libraries unless you update your environment.

   * SHLIB_PATH is the shared library definition for the HP OS
   LD_LIBRARY_PATH is the shared library definition for the Solaris OS
   LIBPATH is the shared library definition for the Aix OS

```
$ pwd
/home/jtest
$ mkdir TDClient
$ cd TDClient
$ ll
total 12874
-rwxrw--w-   1 jtest    users      6590934 May 20 11:46
tdaccess_v32_FSA_HPUX11.SFX
$ ^C
$ -rwxrw--w-   1 jtest    users      6590934 May 20 11:46
tdaccess_v32_FSA_HPUX11.SFX
ksh: -rwxrw--w-:  not found
$ tdaccess_v32_FSA_HPUX11.SFX
TDCompress Build 0465 (master, triple DES) (c) Copyright 1990-2003
DECOMPRESS STARTED - Tue May 20 11:51:00 2011

Decomping tdaccess_v32_FSA_HPUX11.SFX
Decomped  install.dat
Decomped  tdsetup.ksh

Installing TDAccess 3.2, containing TDClient 3.2 and TDServer 3.2 ...

Please enter the directory in which TDAccess
is to be installed  [/home/jtest/TDAccess3.2]

Directory /home/jtest/TDAccess3.2 does not exist

Decomped  ./libtdclients.sl
Decomped  ./libtdnmib.sl
Decomped  ./license.txt
Decomped  ./listrtm
Decomped  ./outmsgp
Decomped  ./pfxhelp.txt
Decomped  ./readme.inv
Decomped  ./readme.txt
Decomped  ./release.txt
Decomped  ./run_in_background.ksh
Decomped  ./tdclient.cmd
Decomped  ./tdclient_example_read.me
Decomped  ./tdclientc
Decomped  ./tdserver
Decomped  ./tdserver.cfg
Decomped  ./tdserver_install_guide.doc
Decomped  ./tpaddrss.ini
*.ini not found
Creating TDAccess subdirectories...
Resetting file permissions for TDAccess files...
cp -p exfer.ini /home/jtest/TDAccess3.2/exfer.ini
cp -p tpaddrss.ini /home/jtest/TDAccess3.2/tpaddrss.ini
Installation completed!
```

Figure 5: Example of install text displayed on screen

Note: You may need to modify the .profile record of the individual(s) who will be running TDClient. Add a variable called "LD_LIBRARY_PATH= ", "SHLIB_PATH=", or

"LIBPATH=", depending on your UNIX/LINUX OS, to define the path to the TDC directory. There are library files contained in the TDC directory that are needed by the TDClient executable.

2.4 Installation on OS/400 System

To use TDClient, you must have OS/400 v5R4or 6.1+. You must also have installed and configured the TCP component of OS/400 and establish a physical connection to the internet.

2.4.1 Materials required prior to installation:

- Download the OS/400 TDClient from the FSA Download website https://www.fsadownload.ed.gov/softedconnect.htm. TDCLNTSAV
- Download the TDClient.INI file from **https://www.fsadownload.ed.gov/softedconnect.htm**

2.4.2 Installing TDClient OS/400 Software:

1. Create an empty save file on the OS/400.

   An example command to type would be - **CRTSAVF SAVEFILE.**

2. Upload the TDCLNTSAV file from the Windows PC to the new OS/400 SAVE file using a binary mode FTP transfer. See Figure 6 below.

```
> ftp x.x.x.x (x.x.x.x = IP Add)        connect to OS/400 at network
address

220 User (OS/4000:(none)): userid        type userID necessary

331 Enter password.                      type password needed

230 USERID logged on.

ftp> bin                                 switch to binary transfer mode

220 Representation type is binary IMAGE.  confirmation

ftp> put TDCLNTSAV SAVEFILE      transfer installation save file

200 PORT subcommand request successful.

150 Sending file to member SAVEFILE in file SAVEFILE.

250 File transfer completed successfully.

ftp> quit                                end FTP session
```

Figure 6: Example FTP Session to Transfer TDClient Software to OS/400

3. Use the RSTLIB command to unload the TDClient library. An example command is: **RSTLIB SAVLIB(EA148LIB) DEV(*SAVF) SAVF(SAVEFILE) RSTLIB(EA2KLIB)**

4. After the restore is complete, create the TDCLIENT object: An example command is: **CRTPF FILE(EA2KLIB/TDCLIENT) RCDLEN(200) FILETYPE(*SRC)**

5. FTP the TDClient.INI over to the OS400 in ASCII mode. An example command is: **Put TDClient.INI ea2klib/tdclient rep**

6. Make TDClient programs available to users. An individual user can do this by running “ADDLIBLE newlib” per session or the system administrator can make this available to all users by adding the newlib to the LIBRARY LIST.

2.4.3 Exchanging Data using OS/400 TDClient

1. Use the OS/400 command line or use the TDClient command-line interface application with the name “TDCLIENTC”.

2. Use the PARM keyword on the CALL statement to specify this information:
   - Name of a stored transfer, user ID and password required to logon.
   - Compression and decompression program options or Name of the command file that contains this information.

Note: When using the command file option from the command-line interface, the command file must be in a physical file format and should contain the appropriate transfer, compression, and decompression parameters.

Table 1 displays several command syntax examples used to implement TDClient data transfers on an OS/400 computer.

2.4.4 OS/400 Data Transfers - Command Syntax Examples

Table 1: Command Syntax Examples

Example	Command Examples
Command file name	**CALL** TDCLIENTC) **PARM(‘TRANSFER=CMDFILE=LIBRARY/FILE’)**
Use a stored transfer called from command-line interface.	**CALL** TDCLIENTC) **PARM(‘TRANSFER=TRANSFER’)**
Use a stored transfer with a blank in its name.	**CALL** TDCLIENTC) **PARM(‘TRANSFER=“TEST TRANSFER”’)** Use double quotes to surround the stored transfer name.
Creating a stored transfer for later use.	**CALL** TDCLIENTC) **PARM(‘TRANSFER=NAME=“NEW TRANSFER”** **‘PASSWD=password’ ‘NETWORK=SAIGPORTAL’** **‘ASCII’ ‘CRLF’** **‘COMPRESS’)**

Note: The new transfer name and parameters are appended to the EXFER file and can be called later by using only the stored transfer name. Specifying the transfer and compression options later will not be necessary.

2.4.5 OS/400 Operating System-specific TDClient Considerations

To simplify your command syntax when running utility programs or TDClient data transfers, you may want to use the CHGCURLIB (change current library) command to make the User specific RUNTIME library the current library.

2.4.6 Temporary Work Files

TDClient creates several temporary files as part of its normal application processing. These files are written to the OS/400 designated “current library.” This is another reason for making the user-specific RUNTIME library the current library prior to TDClient execution (with the CHGCURLIB command).

During transmission, TDClient creates temporary files named **SYSUT1** and **SYSUT2** in the current directory. These files hold directory listings and copies of compressed data files. System defaults are usually adequate for creating these temporary files; however, if you send or receive large files you may need to pre-create one or both of the temporary files with an adequate size to hold the data. If this is the case, then create these files as physical files with a record length of 256 bytes. You may need to experiment with the number and size of the record extents to allocate files of the desired sizes.

2.4.7 Naming and Allocating Work Files

Specify the file names used to send and receive data by using the LIBRARY/FILENAME(MEMBER) syntax. If the file is available via the **LIBLIST** command, then you can omit the LIBRARY portion of the command. If the first (or only) library member is needed, then you can omit the (MEMBER) portion of the command.

When receiving data, TDClient creates the output files if they do not exist. However, the files are created in the current library with default values for maximum record length and file size. If the defaults are not acceptable, then you should create the files with the appropriate with number and size of the record extents, prior to receiving the transmitted data.

2.5 Installation on Windows

TDClient provides file transfer capabilities with compression and encryption for the Windows XP, Windows 7, Windows Vista and Windows 2003 and 2008 server platforms.

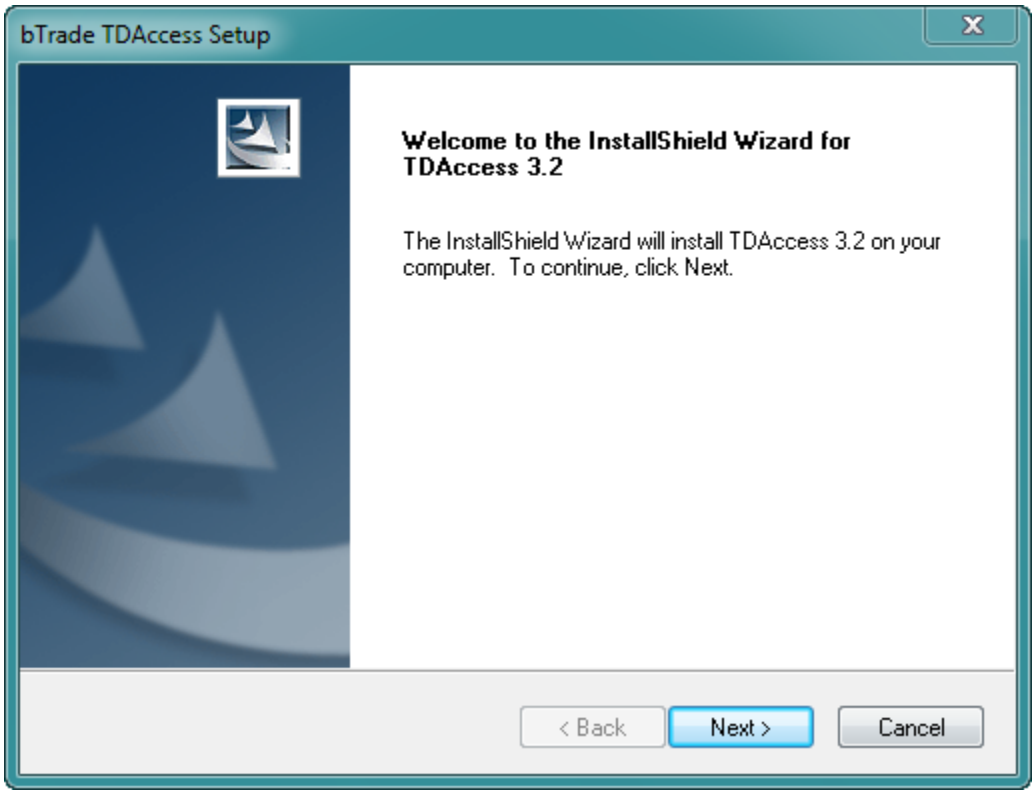
2.5.1 Materials required prior to installation:

- Download the Windows TDClient from the FSA Download website at https://www.fsadownload.ed.gov/softedconnect.htm.
  - **tda3.2_windows.zip**
    - Zip file contains:
      - TDAccess_V32_FSA_windows.exe – to install TDClient
      - See SAIG Host Guide, page 4-4 for instructions on use
- TDCWINEXMP file – the secfile (see SAIG Host Guide, page 4-4 for instructions on use), examples of batch and command line files.The TDClient.INI file and the SECFILE are bundled within the TDAccess3.2_win.zip file and do not need to be downloaded separately.

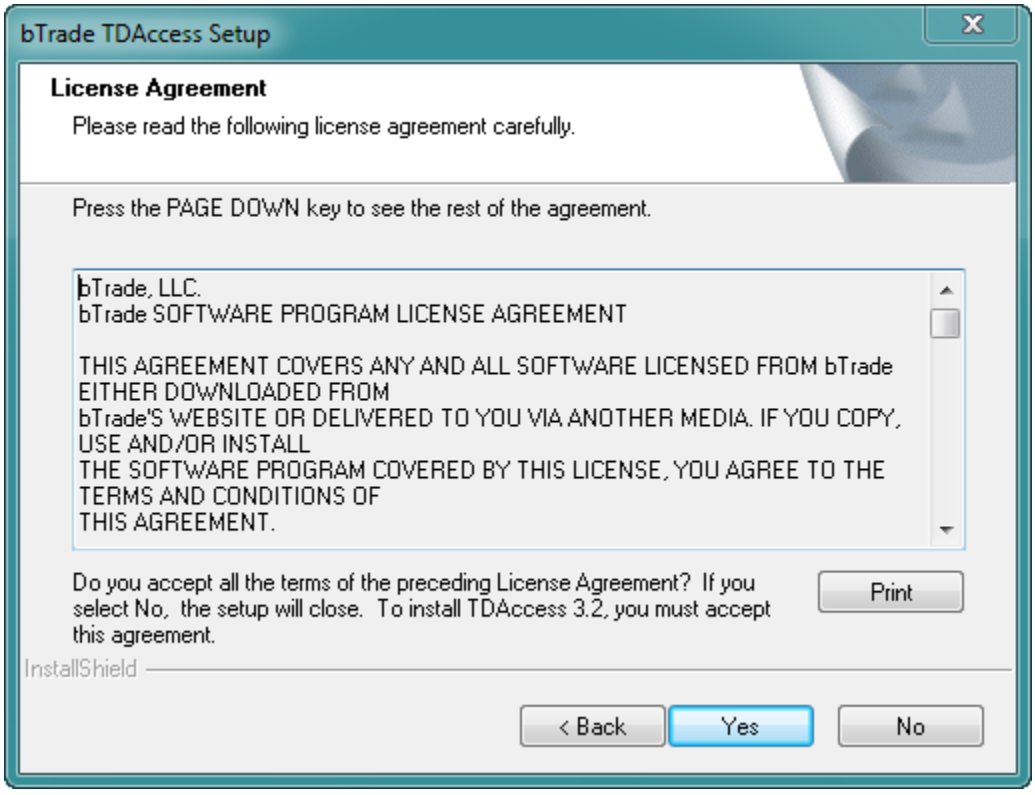
2.5.2 To install TDClient:

1. Unzip the TDAccess3.2_win.zip file to a local directory where you can perform the installation and run the setup.

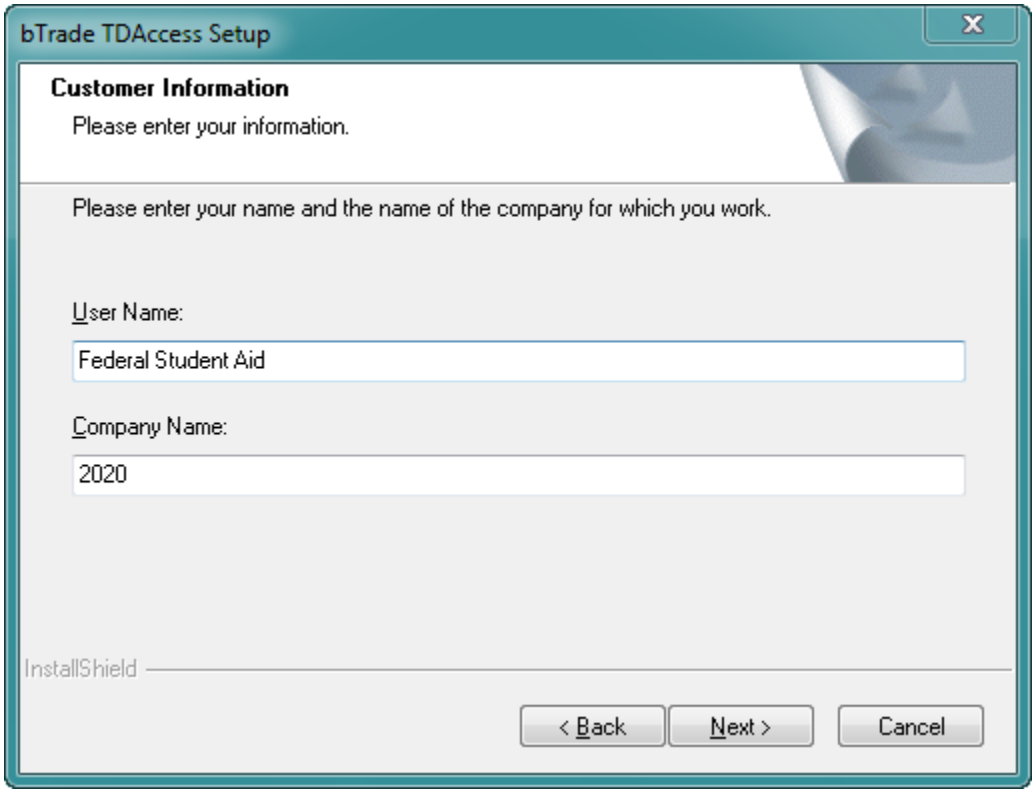
2. To start the Install Shield Wizard, select the setup.exe file.
    a. On the Welcome screen select the ***Next*** button to continue the installation.



b. Read the License Agreement and select the ***Yes*** button to continue the installation or the ***No*** button to cancel the installation.

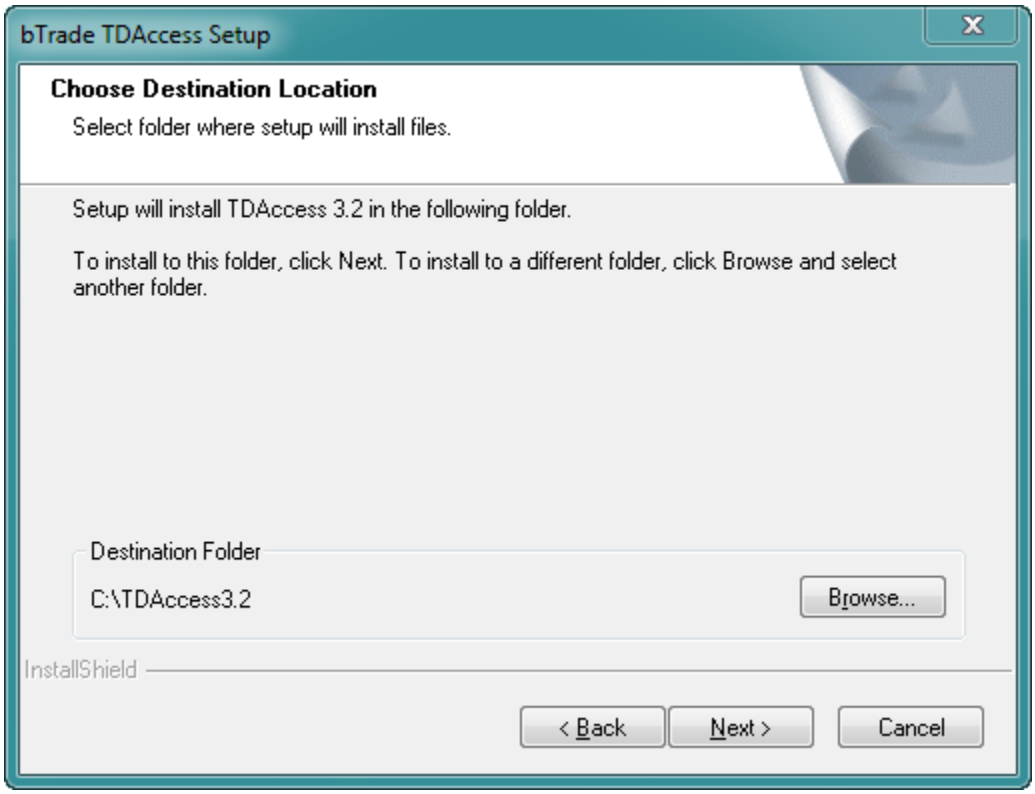


c. Enter the requested Customer Information and select the ***Next*** button to continue the installation.

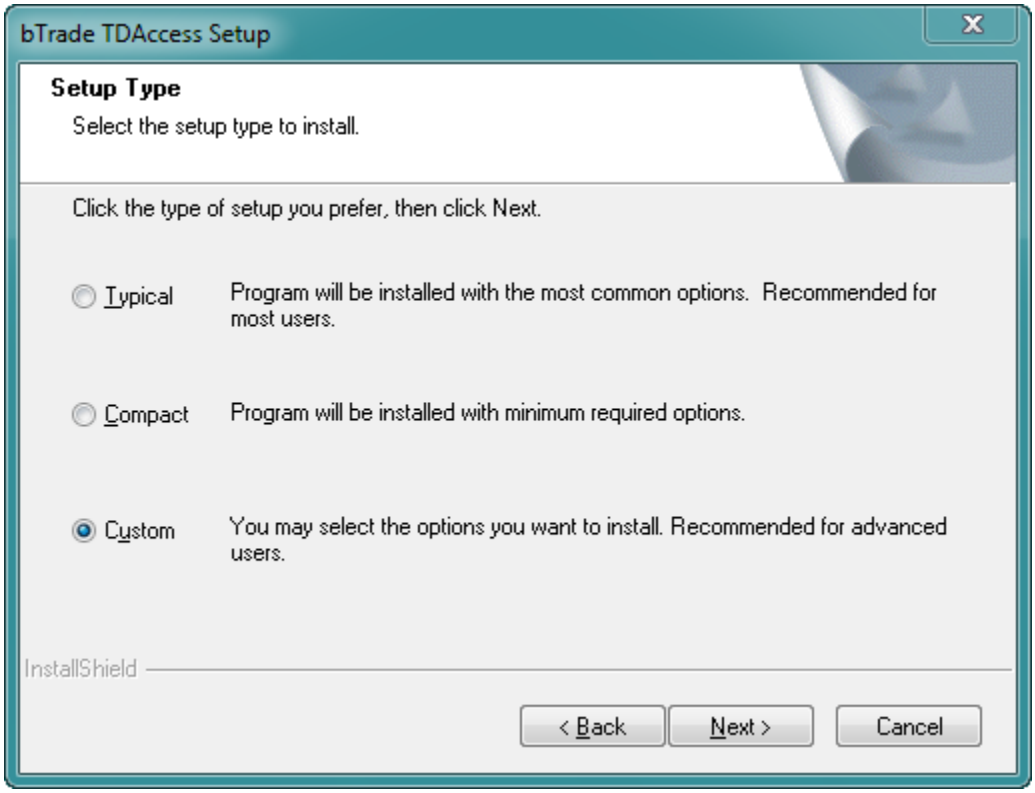


d. Click on Browse to select the folder to install TDClient. Select the ***Next*** button to continue the installation.

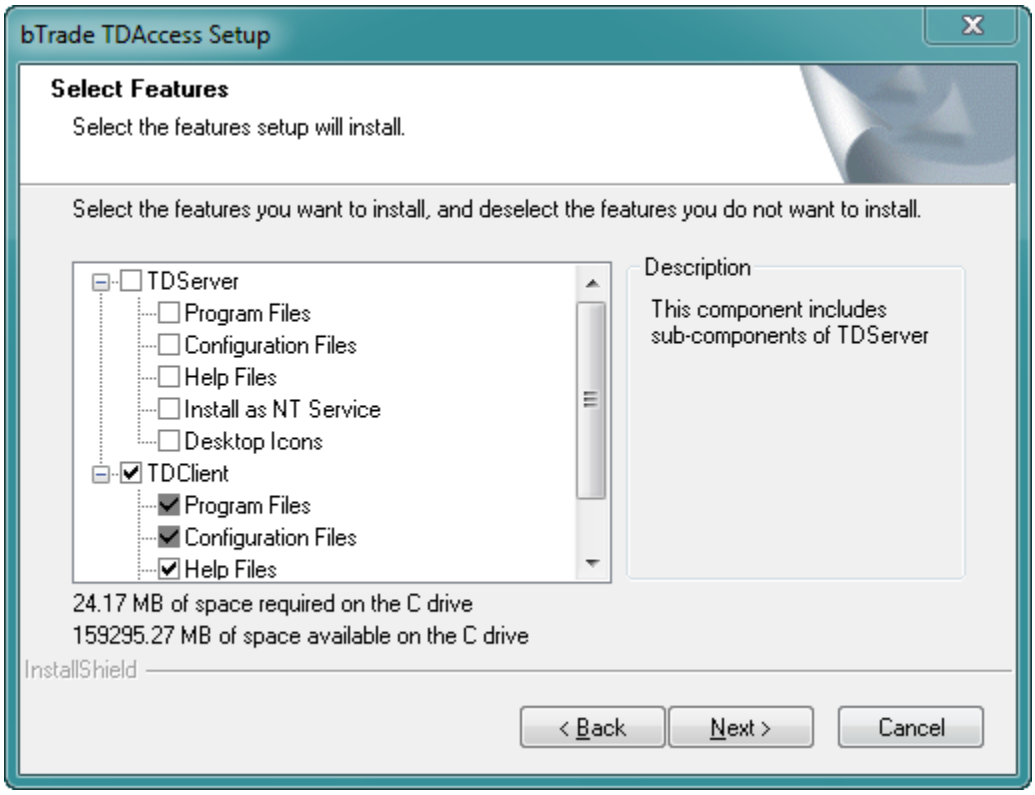
Note: TDClient for Windows version must installed in the **root directory** without any spaces in the path.



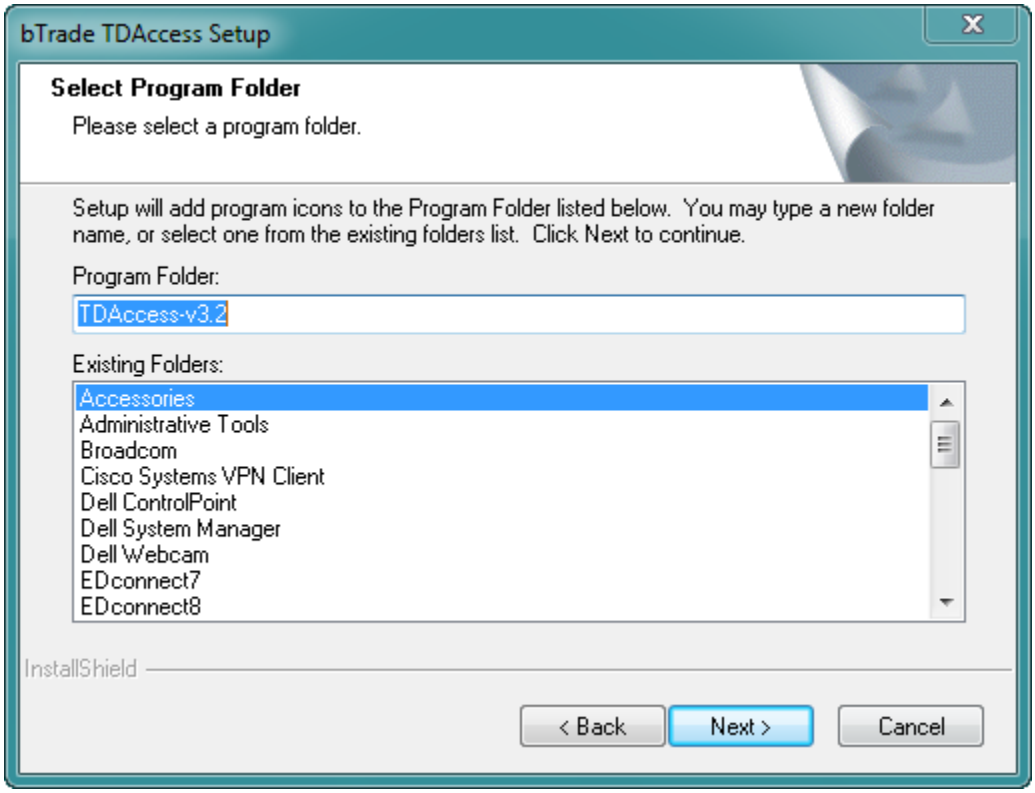
e. Select the ***Custom*** setup and then select the ***Next*** button to continue the installation.



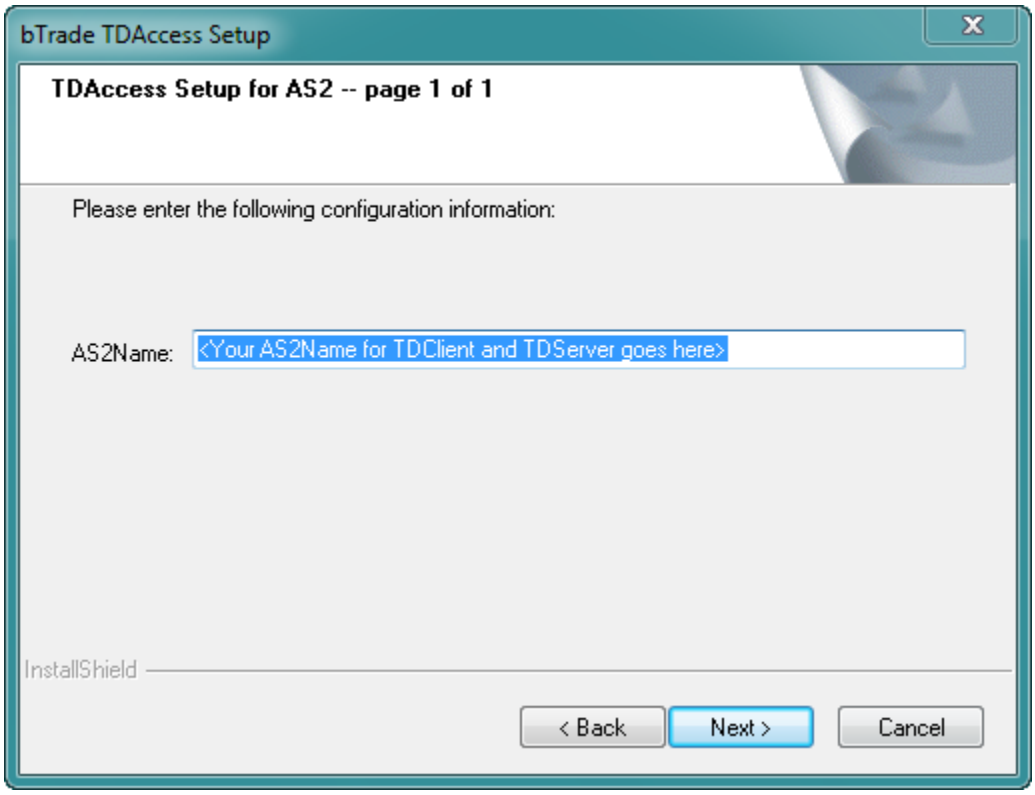
f. Deselect the TDServer folder and sub-components, do select the **TDClient** folder and sub-components and then click on the ***Next*** button to continue with the installation.



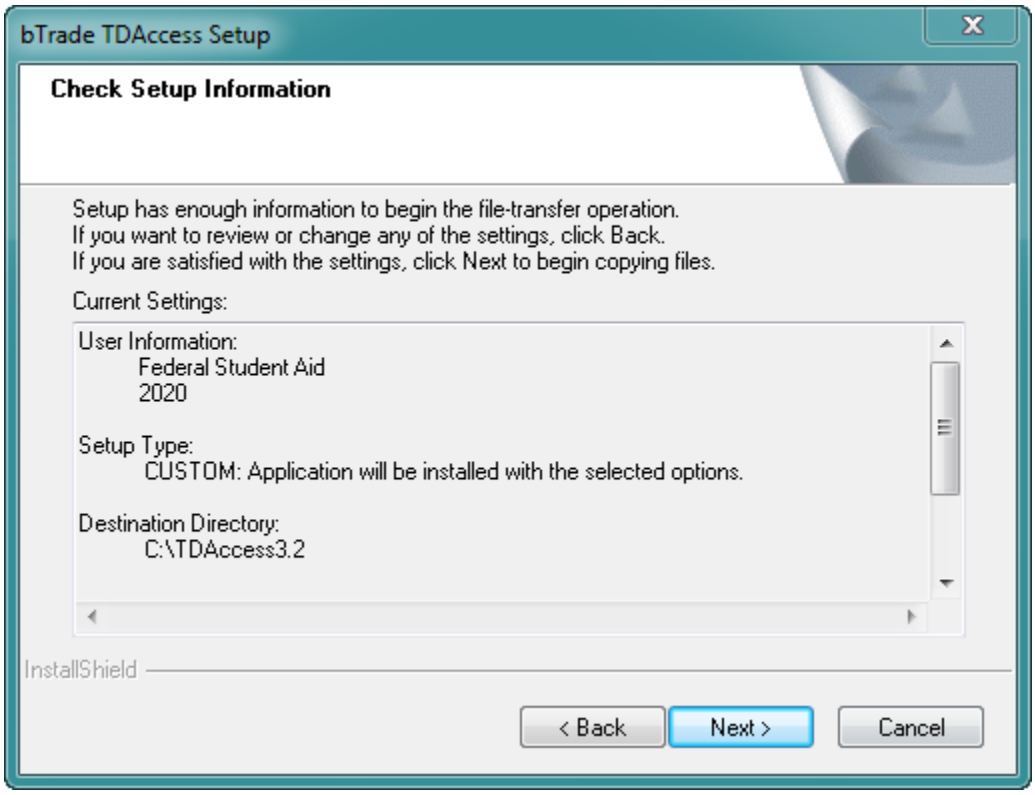
g. Enter a program folder name or use the default of TDAccess-v3.2 and then click on the ***Next*** button to continue with the installation.



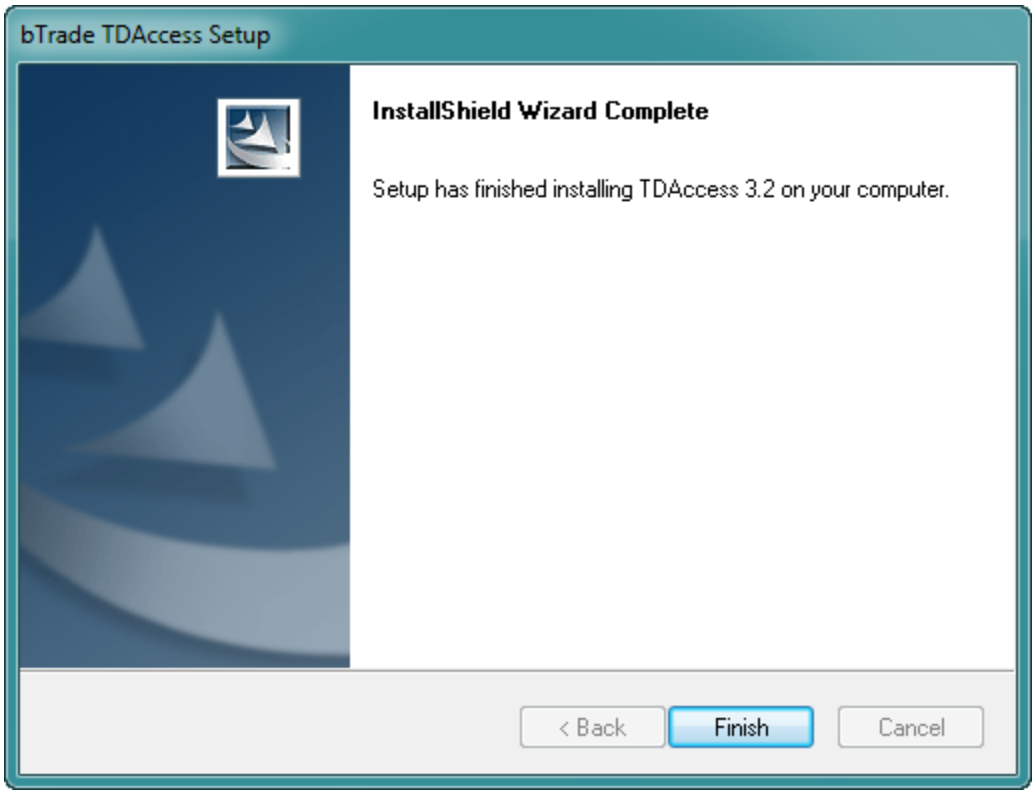
h. Ignore the TDAccess Setup screen and click on the ***Next*** button to continue with the installation.



i. Confirm your setup choices and click on the ***Next*** button to continue with the installation.



j. On the final screen click on the ***Finish*** button to complete the installation.



3 Security

3.1 Password Update Procedure

3.1.1 General Information

If this is a newly enrolled mailbox the initial password for access to your Student Aid Internet Gateway (SAIG) mailbox is set as follows: “ccyynnnn” where “ccyy” is your birth year and “nnnn” represents the last four digits of your social security number. You will be required to change your password the first time you access your mailbox and prior to performing any other activity. (See Figure 3-1.)

The following rules apply when creating a new SAIG network password:

- Password must begin with an alpha character.
- Password must be a minimum length of eight characters.
- Password must contain at least one upper case, one lower case alpha character and one numeric character.
- Password cannot be any of the last five passwords used.
- Password cannot match the User Name or TG number (ignore case).
- Password will be locked out after three failures. (Password will be unlocked after 30 minutes or you can call CPS/SAIG Technical Support to have the password reset.)
- Passwords expire every 90 days, but you can change your password more frequently.

If you have any difficulty establishing your first password, contact CPS/SAIG Technical Support at **800/330-5947** and request to have your SAIG mailbox password reset.

Note: Network passwords can be changed by either of two methods: via batch job with the TDClient software, or with the TDCommunity Manager (TDCM). This document addresses only the batch process. The TDCM User Manual containing instructions on the alternate method of updating your password is available on the FSAdownload Web site.

3.2 Batch Procedure

There are two methods for changing your network password. The currently used FTPPASSWD parameter shows how to change your mailbox password using any version of TDClient. The new parameter, NEW_PASSWD, can be used with newer versions of TDClient version 2.2 and above. A password change cannot be submitted by itself and must be accompanied by some other network activity such as, sending data, receiving data or a query list.

We recommend the following procedure for changing passwords every 90 days:

1. Create a separate job for password changes only.

2. If using the FTPPASSWD parameter to change your password (i.e., FTPPASSWD=oldpass/newpass/newpass) then, after changing your network password, run another job to send or receive a file using the new password in your command line. Submission of the new password will update the stored password that is currently in the TDCLIENT.INI file. The stored password is encrypted for security and will not change unless you submit another password change.

   Note: We recommend using the NEW_PASSWD parameter if using TDClient version 2.2 or higher.

3. Since TDClient will store any password supplied on the command line, we suggest that you maintain a separate TDCLIENT.INI file for testing purposes.

4. Remove the FTPPASSWD= parm from all existing programs and JCL. Removal of this parameter will force TDClient to use the encrypted password that is stored in the TDCLIENT.INI file.

If your SAIG password expires you will receive the following error in your SYSOUT file or logs:

- WARNING: Logon to server failed
- Login for UserID: TG71504 failed
- (531) FTP login failed. 531 Change password required

The data elements required as input to the batch password update using the FTPPASSWD parameter: **FTPPASSWD=oldpass/newpass/newpass**

FTPPASSWD=

- Old password: 8 characters
- Forward slash separator: /
- New password: 8 characters
- Forward slash separator /
- Verify password: 8 characters

Note: No spaces between the old password, forward slash, and the new password.

The data elements required as input to the batch password update using the NEW_PASSWD parameter: **NEW_PASSWD=newpass**

NEW_PASSWD=

- New password: 8 characters

Note: We recommend using the NEW_PASSWD parameter. Use of the NEW_PASSWD=password causes the TDClient to change the network password on the server and will update the encrypted password which is stored in an encrypted form in the TDClient.INI file. Multiple transmissions will not be necessary.

Note: Use with MVS zOS TDClient version 2.2 or higher to change the network password and to update the tdclient.ini file with the new password.

```
//CMDSEND   DD *
 NETWORK=SAIGPORTAL FTPUSERID=TGxxxxx
 NEW_PASSWD=newpassword RESET
 QUERY_LIST QUERY_FILE=DD:QUERYfilename
```

Figure 7: Example MVS z/OS Password Change

Note: Use with older versions of MVS z/OS TDClient to change the network password only.

```
//CMDSEND   DD *
 NETWORK=SAIGPORTAL FTPUSERID=TGxxxxx
 FTPPASSWD=oldpass/newpass/newpass RESET
 QUERY_LIST QUERY_FILE=DD:QUERYfilename
```

Figure 8: Example MVS z/OS Password Change (Older Versions)

Note: Use with OS/400 TDClient to change the network password only.

```
Chgpasswd CMDFILE

crtpf file(ea148lib/passchg) rcdlen(80) filetype(*src)
*************** Beginning of data **********
NETWORK=SAIGPORTAL
FTPUSERID=TGxxxxx
FTPPASSWD=OLDPASSWORD/NEWPASSWORD/NEWPASSWORD RESET
QUERY_LIST QUERY_FILE=DD:QUERYfilename
****************** End of data *************
```

Figure 9: Example OS/400 Command Line for Password Change

TDClient command line options for UNIX/Linux are NOT case sensitive. Backslashes are being used at the end of each line for line continuation. Double quotes or no quotes can be used in command lines that use the backslashes for line continuation. Do not use single quotes. Change the executable name in the example to the appropriate UNIX client that you have installed.

Note: Use with TDClient version 2.2 or higher to change the network password and to update the tdclient.ini file with the new password.

```
tdclientc "network=saigportal" ftpuserid=TGxxxxx          \
NEW_PASSWD=NEWPASSWORD reset query_list
```

Figure 10: Example UNIX/LINUX Script for Password Change

Note: Use with older versions of TDClient to change the network password only.

```
tdclientc "network=saigportal" ftpuserid=TGxxxxx          \
ftppasswd=OLDPASSWD/NEWPASSWD/NEWPASSWD reset query_list
```

Figure 11: Example UNIX/LINUX Script for Password Change (Older Versions)

```
tdclientc RESET "network=saigportaldev" ftpuserid=TGxxxxx \
ftppasswd=PASSWORD query_list save_only
```

Figure 12: Example UNIX/LINUX Script to Synchronize INI File Password with Current Network password

```
tdclientc RESET "network=saigportaldev" ftpuserid=TGxxxxx
ftppasswd=PASSWORD query_list save_only
```

Figure 13: Example Windows Command Line to Synchronize INI File Password with Current Network Password

```
tdclientc RESET "network=saigportal" ftpuserid=TGxxxxx
NEW_PASSWD=PASSWORD query_list
```

Figure 14: Example Windows Command Line to Change Network Password and Update the INI File

Note: See sample batch files in the TDAccess3.2_win.zip file, in the TDCWINEXMP folder.

```
@ECHO OFF
REM This batch file is used to set the TDCLIENT.INI id and password
w/o a network change
REM it can also be used to sync a pwd when a TDCM change was done.
REM enter ID and Password on command line

:: Set default ID, PASSWORD
SET ID=TGXXXXX
SET PSWD=PASSWORD

:: Use command-line settings if given
IF NOT (%1)==() SET ID=%1
IF NOT (%2)==() SET PSWD=%2

cd ..
@ECHO ON
tdclientc RESET "network=saigportaldev" ftpuserid=%ID%
ftppasswd=%PSWD% query_list save_only
cd maint
```

Figure 15: Example Windows Batch File to Synchronize INI File Password with Current Network password

Note: See sample batch files in the TDAccess3.2_win.zip file, in the TDCWINEXMP folder.

```
@ECHO OFF
REM network password change
REM Assumes tdclient.ini and mailbox password are in sync
REM If not use TDCset.bat to sync tdclient.ini and mailbox password
REM
REM Can enter id & pwd on command line or it will use defaults set below

:: Set default ID, PASSWORD
SET ID=TGXXXXX
SET PSWD=PASSWORD

:: Use command-line settings if given
IF NOT (%1)==() SET ID=%1
IF NOT (%2)==() SET PSWD=%2

cd ..
@ECHO ON
tdclientc RESET "network=saigportal" ftpuserid=%ID% NEW_PASSWD=%PSWD%
query_list

cd maint
```

Figure 16: Example Windows Batch File to Change Network Password and Update the INI File

4 Communication Procedures

4.1 Introduction

This section describes basic procedures for sending and receiving data over the Student Aid Internet Gateway (SAIG).

This section:

- Contains a list of keywords
- Provides examples for sending and receiving data
- Describes the Query List function used to manage the contents of your mailbox(es)
- Documents the record layouts for network headers and trailers

4.2 Command Line Keywords

Command line keywords control the login process as well as what files are to be sent or received. The same commands (keywords) are used on all platforms. The appropriate transfer command lines are combined with the network command line to perform the desired actions as shown in Figure 4-1.

```
NETWORK=SAIGPORTAL FTPUSERID=TGxxxxx RESET
TRANSFER=(NAME=yourname SENDUSERID=TGxxxxx
SEND=:inputfilename
      OTHER_COMP_PARMS='SECFILE=secfilename')
```

Figure 17: Example Network and Transfer Command Lines for Sending a Single File

4.2.1 Network Command Line Keywords

Table 2 lists keywords used in the NETWORK command line to sign on to TDPortal and perform any network activity: sending, receiving, query list or password changes.

Table 2: Network Command Line Keywords

Keyword	Definition
NETWORK	NETWORK=SAIGPORTAL is required. This parameter defines the secure network being used for the communications session. This will always be SAIGPORTAL to match the network defined in your TDCLIENT.INI file.
FTPUSERID	FTPUSERID=TGxxxxx is required (where "xxxxx" is your five digit TG number.) This is your SAIG mailbox user ID. This will not change from what you currently use.
FTPPASSWD	This is the password associated with the FTPUSERID. The TDCLIENT.INI file has the default set to CASE= (blank, to send case sensitive data to the server unchanged). **Note: Once you have reset your password it is stored in the TDCLIENT.INI file in encrypted format and you will no longer need to use the FTPPASSWD parameter in your programs and JCL. See Section 3 for changing and maintaining passwords.**
RESET	Instructs TDClient to ignore any previously failed Transfers which would otherwise attempt to restart.
SAVE	Instructs TDClient to save the data you specify on the command-line or in the command-file in the TDCLIENT.INI file, causing the data to be permanently in effect until you change it.
SAVE_ONLY	Acts like the SAVE keyword, except the program exits after saving the specified data in the TDCLIENT.INI file. **Note: only for versions 2.2 and above.**

4.2.2 Transfer Command Line Keywords for Sending Data

Table 3: Transfer Command Line Keywords for Sending Data

Keyword	Definition
TRANSFER	This defines the transfer parameters of data being sent. Up to 200 transfers can be CREATED within the command-file. **Note:** A TRANSFER=NAME= can be saved to the TDCLIENT.EXFER.INI file by using SAVE at the end of the command line. This is helpful when data of the same message class is sent or received on a routine basis. Once a NAME= is saved, on subsequent job submissions you will only need to specify the saved NAME=, and not any of the other TRANSFER commands. ***TRANSFER contains the following keywords (parameters):***
NAME=	This names the transfer being created. The definition will be saved in the TDCLIENT.EXFER.INI file, provided you use the SAVE command. If the name currently exists it will overwrite the current definition.
SENDUSERID=	SENDUSERID is a required parm in the Transfer command line. You must use a valid TG ID as a place holder in this field. Using your own TG ID is highly recommended. **Note:** Using a TG ID other than your own will cause your job to fail if the TG ID is deleted or inactivated. The O*N05 header record in your data file is the default for the TG ID where the data is being sent.
SEND=	This is the location that the data is being sent from (a UNIX filename or MVS DD name).
SENDCLASS=	SENDCLASS is optional; however, we highly recommend removing this parm from your Transfer command line. The O*N05 header record is the default for the message class. If you use this parm, it will override the default and not use the O*N05 header.
OTHER_COMP_ PARMS=	These are compression parameters used only during the compression step for sending data.
SECFILE=	This is a parameter used in OTHER_COMP_PARMS during sending. It contains the location of the secfile definition. This would be a UNIX filename or MVS DD name in JCL. This required parameter provides two functions: 1) The SECFILE defines the position of each parameter in the network headers and trailers. This information is used by TDPortal to separate files and place files in the correct mailboxes; and 2) It forces TDClient to use the SECFILE parameters to send and receive data properly. See Figure 4-4.
SAVE (Optional)	A TRANSFER NAME can be saved to the TDCLIENT.EXFER.INI file by using this keyword at the end of your command line. This is helpful when data of the same message class is sent or received on a routine basis. Example: *TRANSFER=(NAME=yourname RECEIVE=outputfilename* *RECEIVEUSERID=TGxxxxx RECEIVECLASS=messageclass) SAVE* Once a NAME= is saved, on subsequent job submissions you will only need to specify the saved NAME=, and not any of the other TRANSFER

Keyword	Definition
	commands.
SAVE_ONLY (Optional)	Acts like the SAVE keyword, except the program exits after saving the specified data in the TDCLIENT.INI file. **Note: only for versions 2.2 and above.**

```
HEADERLITERAL(O*N05) HEADERSTART(1) RECEIVERSTART(6) RECEIVERLENGTH(14)
CLASSSTART(25) CLASSLENGTH(8);
TRAILERLITERAL(O*N95) TRAILERSTART(1);
```

Figure 18: Example SECFILE

Note: Pay close attention to the placement of semicolons and spaces, or errors August result. The SECFILE is required when sending data.

- HEADERSTART(1) indicates the O of the O*N05 to start in the first position.
- RECEIVERSTART(6) indicates that the receiver ID of a file being sent begins in the 6th position of the N05 header.
- RECEIVERLENGTH(14) indicates the length of the receiver ID field, including spaces.
- Using the CLASS options in the SECFILE forces TDClient to use the message class (CLS=) contained in the O*N05 header record.
  - CLASSSTART(25) indicates that the message class begins in the 25th position of the N05 header.
  - CLASSLENGTH (8) indicates that the CLS= field is 8 positions long.
- All SECFILE parameters referring to Headers and Trailers that define the network headers and trailers are required.
  - The O*N05 Transmission Header and the O*N95 Transmission Trailer surround each set of application system headers and trailers in the data file being sent.

Note: See Appendices A and B for more information on use of the TRANSFER command to control the data you send or receive.

- Appendix A, Example 2, is an example of how to compress a file separately from TDClient prior to sending the file.
- Appendix A, Example 3, is an example of how to send an already compressed file with compression turned off in TDClient.

```
//STEP0020  EXEC PGM=EA2KMVSC,REGION=4M,TIME=1000,
//          PARM='CMDFILE=DD:CMDSEND'
//STEPLIB   DD DSN=your.dataset.prefix.TDLOAD,DISP=SHR
//*
//EASYACC   DD DSN=your.dataset.prefix.TDCLIENT.INI,DISP=SHR
//*
//EXFER     DD DSN=your.dataset.prefix.TDCLIENT.EXFER.INI,DISP=SHR
//*
//CMDSEND   DD *
NETWORK=SAIGPORTAL FTPUSERID=TGxxxxx RESET
TRANSFER=(NAME=yourname SENDUSERID=TGxxxxx SEND=DD:SENDFILE
OTHER_COMP_PARMS='SECFILE=DD:SECFILEX')
//*
//SENDFILE   DD DSN=your.send.file,DISP=SHR
//*
//SYSUT1    DD DISP=NEW,UNIT=SYSDA,SPACE=(CYL,(5,5)),
//             LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK01    DD DISP=NEW,UNIT=SYSDA,SPACE=(CYL,(5,5)),
//             LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK02    DD DISP=NEW,UNIT=SYSDA,SPACE=(CYL,(5,5)),
//             LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK03    DD DISP=NEW,UNIT=SYSDA,SPACE=(CYL,(5,5)),
//             LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK04    DD DISP=NEW,UNIT=SYSDA,SPACE=(CYL,(5,5)),
//             LRECL=8192,BLKSIZE=0,RECFM=VB
//EASTATUS  DD DSN=your.EASTATUS,
//             DISP=(NEW,CATLG),UNIT=SYSDA,SPACE=(CYL,(5,5)),
//             LRECL=8192,BLKSIZE=0,RECFM=VB
//SECFILEX  DD *
SENDER(TGxxxxx);
HEADERLITERAL(O*N05) HEADERSTART(1) RECEIVERSTART(6) RECEIVERLENGTH(14)
CLASSSTART(25) CLASSLENGTH(8);
TRAILERLITERAL(O*N95) TRAILERSTART(1);
LITERAL(O*N01) LITERALSTART(1) DROP(Y);
LITERAL(O*N99) LITERALSTART(1) DROP(Y);
//*
//OUTMSG    DD SYSOUT=*
//SYSPRINT  DD SYSOUT=*
//EAFTPLOG  DD SYSOUT=*
//EALOG     DD SYSOUT=*
//EXFERLOG  DD SYSOUT=*
//COMPLOG   DD SYSOUT=*
//CPFTPLOG  DD SYSOUT=*
```

Figure 19: Example MVS JCL to SEND Data

Note: Insert your own dataset names, TG numbers, and time parameters as appropriate.

```
tdclientc network=saigportal ftpuserid=TGxxxxx reset \
"transfer=(name=yourname                                          \
 senduserid=TGxxxxx                                                \
 send=/your/send/file.txt                                           \
 other_comp_parms=secfile=./path/to/your/secfile.txt)"
```

Figure 20: Example UNIX/LINUX Script to SEND Data

Note: Replace the tdclientc with the appropriate client name. Backslashes are being used at the end of each line for line continuation. Double quotes or no quotes can be used in command lines that use the backslashes for line continuation. Do not use single quotes. You can have more than one "transfer=(DATA)" line to transmit multiple files.

See **Figure 18** for example of SECFILE.

```
crtpf file(ea148lib/ncssend) rcdlen(80) filetype(*src)

*************** Beginning
NETWORK=SAIGPORTAL
FTPUSERID=TGxxxxx
RESET
TRANSFER=(NAME=yourname
SEND=EA148LIB/FILE
SENDCLASS=CLASS
OTHER_COMP_PARMS='SECFILE=EA148LIB/SECFILEname')
****************** End
```

Figure 21: Example OS/400 Commands to SEND Data

1. Create a physical file for the command parameters and a physical file for the secfile.

   ```
   CRTPF FILE(EA2KLIB/CMDFILE) RCDLEN(80) FILETYPE(*SRC)

   CRTPF FILE(EA2KLIB/SECFILE) RCDLEN(80) FILETYPE(*SRC)
   ```

2. Write the transfer parameters to the command file and the sender receiver characteristics for the secfile. See **Figure 18** for example of SECFILE.

Note: See sample batch files in the TDAccess3.2_WIN.zip file, in the TDCWINEXMP folder.

```
tdclientc network=saigportal ftpuserid=TGxxxxx reset
"transfer=(name=yourname senduserid=TGxxxxx send=\your\send\file.txt
 other_comp_parms=secfile=.\path\to\your\secfile.txt)"
```

Figure 22: Example Windows Command Line to SEND Data

Note: See sample batch files in the TDAccess3.2_WIN.zip file, in the TDCWINEXMP folder.

```
TRANSFER=(
    NAME=TDCSend
    senduserid=TGXXXXX
    SEND=outgoing\goodsend.txt
    OTHER_COMP_PARMS='SECFILE=maint\secfile'
) SAVE
```

Figure 23: Example Windows Batch File to SEND Data

Note: See sample batch files in the TDAccess3.2_WIN.zip file, in the TDCWINEXMP folder.

```
@ECHO OFF
REM Executes transfer statements in the .c files

:: Set default command file.
SET ID=send.c

:: Use command-line settings if given
IF NOT (%1)==() SET FILE=%1

cd ..
@ECHO ON
TDCLIENTc.exe "network=saigportal" RESPLOG=TEMP\RESPLOG.TXT
CMDFILE=MAINT\TRANS\%FILE% RESET
cd maint
```

Figure 24: Example Windows Batch File to Transmit Data

```
O*N05TGxxxxx       ,CLS=messclas,XXX,BAT=xxxxxxxxxxxxxxxxxxxxxxxxx,
CPS HEADER  0450HTG51234  20000718150719            0001#C10025002000071
10024720300'01002SAM                          008002472031
CPS TRAILER 0450H       20000718150719            000100010#C100250020
O*N95TGxxxxx       ,CLS=messclas,XXX,BAT=xxxxxxxxxxxxxxxxxxxxxxxxx,
```

Figure 25: Example of Send File with Network Headers

See **Figure 17** for parameters SENDUSERID and SENDCLASS in the Command Line Input section. See end of section 4, *Header and Trailer Record Layouts*, for specifics in creating O*N05 and O*N95 headers and trailers.

Following are lists of Input, Temporary and Output logs for send and receive JCL.

4.2.3 Input Logs

Table 4: Input Logs

Input DD	Definition
STEPLIB	Dataset name (your.dataset.prefix.TDLOAD) containing the TDClient program libraries you installed.
EASYACC	Dataset name containing the TDCLIENT.INI file, which contains network access information. – *Do NOT alter this file*.
EXFER	Dataset name containing the TDCLIENT.EXFER.INI file, which stores your saved TRANSFER commands.
CMDSEND*	**Command and Transfer statements to Send data.**
CMDRECV**	**Command and Transfer statement to Receive data.**
SENDFILE*	Location of the input file you want to send from your SAIG mailbox.
RECVFILE**	Location of the pre-allocated files that will receive data pulled from your SAIG mailbox.

*** Used only on Send Transmissions**
**** Used only on Receive Transmissions**

4.2.4 Temporary Logs

Table 5: Temporary Logs

Temporary DD	Definition
SYSUT1***	Holds directory listings and copies of compressed data files.
WORK01***	Receives compressed data and decompresses into the Receive file. If using COMPRESS=N then this file is not used.
WORK02***	Temporary storage for Query_List.
WORK03***	Not used
WORK04***	Works in conjunction with Eastatus
EASTATUS	Contains any errors during a send or receive session. **Must be an allocated file; cannot be a temporary file.**
DCMPLOG**	Logs decompression step for each file received. Indicates if any files failed decompression.

4.2.5 Output Logs

Table 6: Output Logs

Output DD	Definition
SECFILE	This name comes from the "SECFILE=secfilename" in the TRANSFER statement. It stores the SECFILE command lines that tell TDClient how to format the headers and trailers during compression/decompression.
OUTMSG	Confirms successful: a) login, compression, and send of file, or b) decompression and receipt of files.

Output DD	Definition
SYSPRINT	Logs the stored Transfer processing.
EAFTPLOG	Verifies success or failure of logging onto system and send/receive of files as identified by unique filename. Logs all internal and external FTP activities.
EALOG	General log of the session.
EXFERLOG	Verifies all internal FTP, compression, and decompression activities.
COMPLOG*	Verifies successful compression of data.
CPFTPLOG	Log of all commands and responses to and from the FTP server that is normally used for trouble shooting purposes

*** Used only on Send Transmissions**
**** Used only on Receive Transmissions**
***** Temporary work files required by the TDClient software. They can be defined as temporary files with the following parameters: LRECL= 8192, RECFM=VB, BLKSIZE=0.**

4.2.6 UNIX/LINUX and Windows Output Logs

Table 7: Output Logs

Filename	Definition
STDOUT	Confirms successful: a) login, compression, and send of file, or b) decompression and receipt of files.
EA2K.LOG	Verifies success or failure of logging onto system and send/receive of files as identified by unique filename. Logs all internal and external FTP activities.
EACOMM.LOG	General log of the session.
EAXFER.LOG	Verifies all internal FTP, compression, and decompression activities.
COMPRESS.LOG	Verifies successful compression of data.
FTPLOG.TXT	Log of all commands and responses to and from the FTP server that is normally used for trouble shooting purposes

4.2.7 Transfer Command Line Keywords for Receiving Data

Table 8: Transfer Command Line Keywords for Receiving Data

Keyword	Definition
TRANSFER	This defines the transfer parameters of data being received. Up to 200 transfers can be CREATED within the command-file. **Note:** A TRANSFER=NAME= can be saved to the TDCLIENT.EXFER.INI file by using SAVE at the end of the command line. This is helpful when data of the same message class is sent or received on a routine basis. Once a NAME= is saved, on subsequent job submissions you will only need to specify the saved NAME=, and not any of the other TRANSFER commands. ***Within this keyword are the following keywords (parameters):***
NAME=	This names the transfer being created. The definition will be saved in the EXFER.INI file, provided you use the SAVE command. If the name

Keyword	Definition
	currently exists it will overwrite the current definition.
RECEIVE=	This is the location where the data will be received (a UNIX filename or a MVS DD name in JCL).
RECEIVEUSERID= (Optional)	This field contains the mailbox ID you are receiving from, and is optional. If used without RECEIVECLASS, you will receive all data from the specified RECEIVEUSERID.
RECEIVECLASS= (Optional)	This field contains the message class of the data you want to receive, and is optional. If used without RECEIVEUSERID, you will receive all data from the specified RECEIVECLASS. If neither RECEIVEUSERID nor RECEIVECLASS are present, you will receive all data in the mailbox.
OTHER_DECOMP _PARMS= (Optional)	These are decompression parameters used only during the decompression step for receiving data.
SECFILE= (Used to Send files only)	This is a parameter used in OTHER_DECOMP_PARMS during sending. It contains the location of the secfile definition. This would be a UNIX filename or MVS DD name in JCL. This required parameter provides two functions: 1). The SECFILE parses the network headers and trailers used by TDPortal to separate files and place files in the correct mailboxes; and 2). It forces TDClient to use the SECFILE parameters to send data properly. See Figure 4-4.
SAVE (Optional)	A TRANSFER NAME can be saved to the TDCLIENT.EXFER.INI file by using this keyword at the end of your command line. This is helpful when data of the same message class is sent or received on a routine basis. Once a NAME= is saved, on subsequent job submissions you will only need to specify the saved NAME=, and not any of the other TRANSFER commands. Example: *TRANSFER=(NAME=yourname RECEIVE=outputfilename RECEIVEUSERID=TGxxxxx RECEIVECLASS=messageclass) SAVE*
SAVE_ONLY (Optional)	Acts like the SAVE keyword, except the program exits after saving the specified data in the TDCLIENT.INI file. **Note: only for versions 2.2 and above.**

```
NETWORK=SAIGPORTAL FTPUSERID=TGxxxxx RESET
TRANSFER=(NAME=yourname  RECEIVE=outputfilename
RECEIVEUSERID=TGxxxxx RECEIVECLASS=messageclass)
```

Figure 26: Example Network and Transfer Command Lines for Receiving a Specific Message Class from a Specific Sender (RECEIVEUSERID)

You can modify the commands in many different ways, depending on what data you want to receive. To receive all files of a given message class, specify that message class (EAPS02OP, for example) in the RECEIVECLASS command, but do not include the RECEIVEUSERID command.

- To receive all files from a given sender, specify the RECEIVEUSERID, but do not include the RECEIVECLASS command.
- To receive all files in the mailbox, do not include either the RECEIVEUSERID or RECEIVECLASS.
- To receive files of two separate message classes, issue two separate TRANSFER commands, each with separate RECEIVE=DDs or filenames.

Note: See Appendices A and B for more information on use of the TRANSFER command to control the data you send or receive.

- Remember, when receiving files from your mailbox, files are received in the order of the query list option.
- Appendix B, Example 8, is an example of how to receive a file with decompression turned off during the TDClient Receive process.
- Appendix B, Example 9 is an example of how to decompress a file separately from TDClient if the file was received with decompression turned off.

```
//STEP0020  EXEC
PGM=EA2KMVSC,REGION=4M,TIME=20,PARM='CMDFILE=DD:CMDRECV'
//*
//STEPLIB   DD DSN=your.dataset.prefix.TDLOAD,DISP=SHR
//*
//EASYACC   DD DSN=your.dataset.prefix.TDCLIENT.INI,DISP=SHR
//*
//EXFER     DD DSN=your.dataset.prefix.EASYACC.EXFER.INI,DISP=SHR
//*
//CMDRECV DD *
NETWORK=SAIGPORTAL FTPUSERID=TGxxxxx RESET
TRANSFER=(NAME=xxxxxxxx RECEIVE=DD:RECVFLE
   RECEIVEUSERID=TGxxxxx RECEIVECLASS=messageclass)
//*
//RECVFLE   DD DSN=your.dataset.receive.file,
//             DISP=(NEW,CATLG),UNIT=SYSDA,
//             DCB=(LRECL=nnnn,BLKSIZE=nnnnn,RECFM=FB),
//             SPACE=(CYL,(nn,nn))
//*
//SYSUT1    DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//             LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK01    DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//             LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK02    DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//             LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK03    DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//             LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK04    DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//             LRECL=8192,BLKSIZE=0,RECFM=VB
//EASTATUS  DD DSN=your.dataset.prefix.EASTATUS,
//             DISP=(NEW,CATLG),UNIT=SYSDA,SPACE=(CYL,(5,5)),
//             LRECL=8192,BLKSIZE=0,RECFM=VB
//*
//DCMPLOG   DD SYSOUT=*
//*
//OUTMSG    DD SYSOUT=*
//*
//SYSPRINT  DD SYSOUT=*
//*
//EAFTPLOG  DD SYSOUT=*
//*
//EALOG     DD SYSOUT=*
//*
//EXFERLOG  DD SYSOUT=*
//*
//CPFTPLOG  DD SYSOUT=*
//*
```

Figure 27: Example JCL to Receive Data

Note: Insert your own dataset names and TG numbers. When defining the receive file dataset, make sure you have sufficient space allocated and that the record length matches the file you are receiving.

```
tdclientc network=saigportal ftpuserid=TGxxxxx reset \
"transfer=(name=yourname                                    \
receive=./path/to/the/file/to/receive.txt                   \
receiveuserid=TGxxxxx                                       \
receiveclass=xxxxxxxx)"
```

Figure 28: Example UNIX/LINUX Script to Receive Data

Note: Backslashes are being used at the end of each line for line continuation. Double quotes or no quotes can be used in command lines that use the backslashes for line continuation. Do not use single quotes. You can have more than one "transfer=(DATA)" line to transmit multiple files.

```
crtpf file(ea148lib/ncsrecv) rcdlen(80) filetype(*src)
*************** Beginning
NETWORK=SAIGPORTAL
FTPUSERID=TGxxxxx
RESET
TRANSFER=(NAME=yourname
RECEIVE=EA148LIB/RECEIVE
RECEIVECLASS=messageclass)
****************** End
```

Figure 29: Example OS/400 Commands to Receive Data

Note: When receiving a file, the receive file must already exist.

```
 crtpf file(ea148lib/receive) rcdlen(80) filetype(*src)
```

Note: See sample batch files in the TDAccess3.2_WIN.zip file, in the TDCWINEXMP folder.

```
tdclientc network=saigportal ftpuserid=TGxxxxx reset
"transfer=(name=yourname receive=.\path\to\the\file\to\receive.txt
 receiveuserid=TGxxxxx  receiveclass=xxxxxxxx)"
```

Figure 30: Example Windows Command Line to Receive Data

Note: See sample batch files in the TDAccess3.2_WIN.zip file, in the TDCWINEXMP folder.

```
TRANSFER=(
   RECEIVECLASS=GOODSEND
   receive=.\incoming\GOODSEND.0001
 ) SAVE
```

Figure 31: Example Windows Batch File to Receive Data

Note: See sample batch files in the TDAccess3.2_WIN.zip file, in the TDCWINEXMP folder.

```
@ECHO OFF
REM Executes transfer statements in the .c files

:: Set default command file.
SET ID=send.c

:: Use command-line settings if given
IF NOT (%1)==() SET FILE=%1

cd ..
@ECHO ON
TDCLIENTc.exe "network=saigportal" RESPLOG=TEMP\RESPLOG.TXT
CMDFILE=MAINT\TRANS\%FILE% RESET
cd maint
```

Figure 32: Example Windows Batch File to Transmit Data

4.3 Query List & Audit Log

The QUERY_LIST command is used outside of the Transfer statement and supersedes any commands used in the Transfer statement. You can use related keywords, QUERY_FILE= and QUERY_STATUS= to control aspects of this query.

Table 9: Query List and Audit Log

QUERY_LIST	QUERY_FILE=	QUERY_STATUS=	RECEIVE_AUDIT_LOGS
Instructs the client to create and execute a transfer to receive a list of available files from your SAIG mailbox (TGxxxxx).	Specifies the qualified file name of the file to receive the list. If not present, the file list is written to the default file, *list.fil*, in the temp directory.	Specifies that the QUERY_LIST should return a list of files with the following specified statuses only: • AVAILABLE – files that have not been received. • RECEIVED – files that have been received. • DELETED – files that have been deleted.	Instructs the client to create and execute a transfer to receive a list of available files and a list of files that were sent from your SAIG mailbox (TGxxxxx). The status of files in this list are the same as QUERY_LIST and show one additional status called ICFAIL. AUDIT_STATUS= can be used with the values: • ICFAIL - files that have been rejected by the server. • AVAILABLE • RECEIVED • REJECTED – a file whose status has been manually changed on the TDCM web site to “rejected”. The parm AUDIT_TYPE= can be used in conjunction with AUDIT_STATUS, with values of: • SENT – files with a sent status on the TDCM • BOTH – files

QUERY_LIST	QUERY_FILE=	QUERY_STATUS=	RECEIVE_AUDIT_LOGS
			with a sent or received status
Examples: TDclient query_list query_status=available Query_file=c:/temp/list.fil Or TDclient receive_audit_logs			

```
2K.01.43\TG50000\TG40000\SARA03OP\U\4.42o.01\O*N05TG54000
,CLS=SARA03OP,BAT=#E3000000200203150000000,NCT=00000\SENDFILE\2.03
0\ASCCRLFILOTH\29501\TG40000\295011761516330265 81\20010625151607\1\1
523\\A\RECEIVED\20010627153549\0\\\\\\50\50\
```

Figure 33: Example of Query list for RECEIVED status

Note: This is one record of data in a sequential file and each field is delimited with a backslash.

```
V2.2.026\TG51550\TG50002\\U\Win32\\MISSINGHEADER.txt\Bld
04\ASCCRLFILOTH\74915\
TG51550\20080731A00007661156\20080731172151\1\299\\A\ICFAIL
\00000000000000\0\\\\\\50\50\
00000000000000\MSGSPLIT\20080731172154\Missing group header record\
V2.2.026\TG51550\TG51550\CORR04IN\U\Win32\O*N05TG51550
,CLS=CORR04IN,XXX, BAT=MISMATCHED BAT,NCT=00000\MISSINGTRAILER.txt\Bld
04\ASCCRLFILOTH\74917\
TG51550\20080731A00007661160\20080731172201\0\301\\A\ICFAIL
\00000000000000\0\\\\\\50\50\
00000000000000\MSGSPLIT\20080731172204\Missing group trailer record\
```

Figure 34: Example of Audit list for ICFAIL status

The following is *an explanation of each field* of the LIST.FIL file created by TDClient when performing a QUERY_LIST (mailbox list) on the SAIGPORTAL. Each field is separated by a backslash "\".

Note: Refer to Appendix B for transmitting by unique file name.

4.3.1 Description of Fields for a Query or Audit List

Table 10: Description of Fields for a Query or Audit List.

#	Field Name	Description
0	VERSION	EA version; e.g. 2K.01.39 *(Send only)*
1	SENDER	TG number of the Sender (active user)
2	RECVR	TG number of the Receiver (destination)
3	CLASS	Message Class
4	FORMAT	U = Unformatted
5	SYSTYPE	W95 (indicates all Windows systems), AIX, SUN, OS400, compression version of MVS,

#	Field Name	Description
		HPUX. *(Send Only).*
6	O*N05 HEADER	Transmission header, O*N05, information
7	ORIGFILENAME	Temporary Send file name from EDIPDS file for MVS only and the original file name for Unix. *(Send Only)* Or Unique File Number *(Receive Only)*
8 - 9		
10	SESSIONID	EAFTP generated session ID
11	USERID	Userid of user logged in for session
12	UNIQUEFILENAME	EAFTP generated unique filename
13	PUTDATETIME	EAFTP generated (ctime)
14	PUTDURATION	EAFTP generated ?
15	FILESIZE	EAFTP determined
16		
17	CHARFORMAT	A = ASCII, I = BINARY
18	STATUS	Status of file [RECEIVED, AVAILABLE, DELETED, ICFAIL]
19	GETDATETIME	EAFTP generated (ctime)
20	GETDURATION	EAFTP generated
21 - 27		Not used
28	Restore date/time	Used for AUDIT list ONLY
29	Process Failed In	Used for AUDIT list ONLY
30	Fail date/time	Used for AUDIT list ONLY
31	Error message description	Used for AUDIT list ONLY

```
//STEP0020  EXEC PGM=EA2KMVSC,REGION=4M,TIME=100,
//          PARM='CMDFILE=DD:CMDRECV'
//STEPLIB   DD DSN=your.dataset.prefix.TDLOAD,DISP=SHR
//EASYACC   DD DSN=your.dataset.prefix.TDCLIENT.INI,DISP=SHR
//EXFER     DD DSN=your.dataset.prefix.EXFER.INI,DISP=SHR
//CMDRECV   DD *
NETWORK=SAIGPORTAL FTPUSERID=TGxxxxx RESET
QUERY_LIST QUERY_STATUS=AVAILABLE QUERY_FILE=DD:QUERY
//*
//QUERY     DD DSN=your.dataset.name.QUERY.LIST,
//             DISP=(NEW,CATLG),UNIT=SYSDA,
//             DCB=(LRECL=700,BLKSIZE=7000,RECFM=FB),
//             SPACE=(CYL,(30,13)),RETPD=6
//SYSUT1    DD DISP=NEW,UNIT=SYSDA,SPACE=(CYL,(5,5)),
//             LRECL=8192,BLKSIZE=0,RECFM=V
//WORK01    DD DISP=NEW,UNIT=SYSDA,SPACE=(CYL,(5,5)),
//             LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK02    DD DISP=NEW,UNIT=SYSDA,SPACE=(CYL,(5,5)),
//             LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK03    DD DISP=NEW,UNIT=SYSDA,SPACE=(CYL,(5,5)),
//             LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK04    DD DISP=NEW,UNIT=SYSDA,SPACE=(CYL,(5,5)),
//             LRECL=8192,BLKSIZE=0,RECFM=VB
```

```
//EASTATUS  DD DSN=your.dataset.prefix.EASTATUS,
//             DISP=(NEW,CATLG),UNIT=SYSDA,SPACE=(CYL,(5,5)),
//             LRECL=8192,BLKSIZE=0,RECFM=VB,RETPD=6
//OUTMSG    DD SYSOUT=
//SYSPRINT  DD SYSOUT=*
//EAFTPLOG  DD SYSOUT=*
//EALOG     DD SYSOUT=*
//EXFERLOG  DD SYSOUT=*
```

Figure 35: Example Query List JCL

```
TDCLIENTc network=SAIGPORTAL RESET query_list
QUERY_FILE="./maint/query/logs/query_default.txt"
cd maint\query
```

Figure 36: Example UNIX/LINUX Command Line for Query List

```
TDCLIENTc network=SAIGPORTAL RESET query_list
QUERY_FILE =".\maint\query\logs\query_default.txt"
cd maint\query
```

Figure 37: Example Windows Command Line for Query List

```
//STEP0020  EXEC PGM=EA2KMVSC,REGION=4M,TIME=100,
//          PARM='CMDFILE=DD:CMDRECV'
//STEPLIB   DD DSN=your.dataset.prefix.TDLOAD,DISP=SHR
//EASYACC   DD DSN=your.dataset.prefix.TDCLIENT.INI,DISP=SHR
//EXFER     DD DSN=your.dataset.prefix.EXFER.INI,DISP=SHR
//CMDRECV   DD *
NETWORK=SAIGPORTAL FTPUSERID=TGxxxxx RESET
RECEIVE_AUDIT_LOGS
//*
//AUDITLOG     DD DSN=your.dataset.name,
//             DISP=(NEW,CATLG),UNIT=SYSDA,
//             DCB=(LRECL=700,BLKSIZE=7000,RECFM=FB),
//             SPACE=(CYL,(30,13)),RETPD=6
//SYSUT1    DD DISP=NEW,UNIT=SYSDA,SPACE=(CYL,(5,5)),
//             LRECL=8192,BLKSIZE=0,RECFM=V
//WORK01    DD DISP=NEW,UNIT=SYSDA,SPACE=(CYL,(5,5)),
//             LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK02    DD DISP=NEW,UNIT=SYSDA,SPACE=(CYL,(5,5)),
//             LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK03    DD DISP=NEW,UNIT=SYSDA,SPACE=(CYL,(5,5)),
//             LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK04    DD DISP=NEW,UNIT=SYSDA,SPACE=(CYL,(5,5)),
//             LRECL=8192,BLKSIZE=0,RECFM=VB
//EASTATUS  DD DSN=your.dataset.prefix.EASTATUS,
//             DISP=(NEW,CATLG),UNIT=SYSDA,SPACE=(CYL,(5,5)),
//             LRECL=8192,BLKSIZE=0,RECFM=VB,RETPD=6
//OUTMSG    DD SYSOUT=
//SYSPRINT  DD SYSOUT=*
//EAFTPLOG  DD SYSOUT=*
//EALOG     DD SYSOUT=*
//EXFERLOG  DD SYSOUT=*
```

Figure 38: Example RECEIVE_AUDIT_LOGS

Note: Instead of issuing a file name for RECEIVE_AUDIT_LOGS, you will create a DD name called AUDITLOG. The TDClient client is programmed to look for a dataset name called AUDITLOG.

```
TDCLIENTc network=SAIGPORTAL RESET  AUDIT_TYPE=SENT
RECEIVE_AUDIT_LOGS AUDIT_FILE="./maint/query/logs/AUDIT_sent_default.txt"
cd maint\query
```

Figure 39: Example UNIX/LINUX Command Line for RECEIVE_AUDIT_LOGS

```
TDCLIENTc network=SAIGPORTAL RESET AUDIT_TYPE=SENT  RECEIVE_AUDIT_LOGS
AUDIT_FILE=".\maint\query\logs\AUDIT_sent_default.txt"
cd maint\query
```

Figure 40: Example Windows Command Line for RECEIVE_AUDIT_LOGS

4.4 File and Transmission Header & Trailer Record Layouts

The use of ***O*N05 and O*N95 header and trailer records is required.*** Transmission Header (O*N05) and Transmission Trailer (O*N95) records wrap the input data for each destination mailbox and message class. Thus, each transmission will contain a minimum of one header and one trailer record for send files.

The first record in the file is the Transmission Header ('O*N05') record. Your data follows the Transmission Header and after your data the Transmission Trailer ('O*N95') record follows. Transmission Header and Transmission Trailer records identify the input data for each destination mailbox and message class. Figure 4-27 shows two batches of data being sent to TGxxxxx for a message class of MSGCLASS.

Note: All header and trailer records are required to be a minimum record length of 70 characters.

Note: These are examples only and may require customization at your site.

```
O*N05TGxxxxx       ,CLS=MSGCLASS,XXX,BAT=xxxxxxxxxxxxxxxxxxxxxxxxx,
your input data for first batch is inserted here
your input data for first batch is inserted here
your input data for first batch is inserted here
O*N95TGxxxxx       ,CLS=MSGCLASS,XXX,BAT=xxxxxxxxxxxxxxxxxxxxxxxxx,
O*N05TGxxxxx       ,CLS=MSGCLASS,XXX,BAT=,
your input data for second batch is inserted here
your input data for second batch is inserted here
your input data for second batch is inserted here
your input data for second batch is inserted here
O*N95TGxxxxx       ,CLS=MSGCLASS,XXX,BAT=,
```

Figure 41: Data File Transmission Headers & Trailers

*4.4.1 Transmission Header (O*N05) & Trailer Record (O*N95) Layouts (Required)*

The Transmission Header record identifies the beginning of a group of input data records destined for a SAIG mailbox. The Transmission Trailer record identifies the end of this group of records. See Figure 4-28 for the required record layout of the Transmission Headers and Trailers.

The Transmission Header and Transmission Trailer records require these substitutions:

- **Record Identifier –** Use O*N05 for Transmission Header; use O*N95 for Transmission Trailer.
- **Destination Mailbox ID –** The Mailbox ID of who is to receive the data when you are sending; or the Mailbox ID of the sender when you are receiving. See application-specific guides and references for the correct destination mailbox for each message class.
- **CLS=Message Class –** The Message Class of the data you are sending. An eight-character label assigned to a particular type of data by the application system.
- **BAT=, –** The Batch ID or Document ID for the batch you are sending. The parameter "BAT=" and the ending comma is required. (i.e., "BAT=," or if you choose to populate this field with the ID then use up to 50 characters; "BAT=#D300018620030816120145," or "BAT=2003-10-21T16:40:19.3092722120,"

4.4.2 Transmission Header & Trailer Record Layouts

Table 11: Transmission Header & trailer Record Layouts

Column	Length	Entry
1-5	5	Record Identifier *(Required)* Use O*N05 for Transmission Header and O*N95 for Transmission Trailer.
6-12	7	Destination Mailbox ID *(Required)* This field must have the same value on both the header and trailer record.
13-19	7	SPACES *(Required)*
20	1	',' (Comma) *(Required)*
21-24	4	CLS = *(Required)*
25-32	8	Message Class *(Required)* This field must have the same value on both the header and trailer record.
33	1	',' (Comma) *(Required)*
34-37	4	XXX, (Required)
38-41	4	BAT= *(Required)*
42-91 Variable	1-50 Variable	Up to a fifty character batch number. If using all 50 characters then adjust the remainder of the record layout accordingly. If no batch number then comma must follow the equal sign. **Note:** A comma must follow this number. (See next field).
42-92 Variable (Based on length of Batch Number.)	1	',' (Comma) *(Required)* **Note:** Must come directly after the Batch Number. If no batch number then comma must follow the equal sign.
93-EOR Variable, starts after comma.	Variable	The minimum record length required is 70. This can be spaces or any data applicable to your institution (such as NCT=).

The Transmission Header and Trailer records described above must be used with all data. Except for the Record Identifier in positions 1-5, both the **O*N05 and O*N95 records must match exactly from position 6 through the end of the batch number comma that starts in position 43+.**

Note: All header and trailer records are required to be a minimum record length of 70 characters.

Appendix A – Command Lines for Different Methods of Sending Data

1. Sending multiple batches of data in one file requires that you have multiple sets of O*N05 and O*N95 transmission headers and trailers around each batch within the file. You must use the O*N05TGxxxxx in the header and O*N95TGxxxxx in the trailer to specify the destination point for each batch of data. See Chapter 4, *Header and Trailer Record Layouts*, for specifics.

2. Since FTPPASSWD is stored in the TDClient.INI file in encrypted format, it is not necessary to hardcode your password in all of your scripts and/or JCL. See chapter 3 for recommendation to change your password.

Note: These are examples only and may require customization at your site.

Example 1: Sample of multiple Transfer statements to send multiple batches in one file. You must specify a corresponding input file for each Transfer statement.

```
NETWORK=SAIGPORTAL FTPUSERID=TGxxxxx RESET
TRANSFER=(NAME=name1 SENDUSERID=TGxxxxx SEND=FILENAM1
  OTHER_COMP_PARMS='SECFILE=secfilename')
TRANSFER=(NAME=name2 SENDUSERID=TGxxxxx SEND=FILENAM2
  OTHER_COMP_PARMS='SECFILE=secfilename')
```

Example 2: Sample JCL to compress a file prior to the TDClient step. See example 3 to send the compressed file with compression turned off in TDClient.

```
//STEP0010  EXEC PGM=COMPRESS,REGION=4M,TIME=1440,
//          PARM='FILTER ASCII CRLF SECFILE=DD:SECFILX'
//STEPLIB   DD DSN=SAIG.EASYACC.LOADLIB,DISP=SHR
//*
//SYSOUT   DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//COMPLOG  DD SYSOUT=*
//SYSPRINT DD  SYSOUT=*
//DATAIN      DD DSN=SAIG.UNCOMP.EAPS03IN,DISP=SHR
//DATAOT      DD  DSN=SAIG.COMP.EAPS03IN,
//            DISP=(NEW,CATLG),RETPD=30,
//            SPACE=(CYL,(60,60),RLSE),UNIT=SYSDA,
//            LRECL=8192,BLKSIZE=0,RECFM=VB
//*
//SECFILX  DD*
HEADERLITERAL(O*N05) HEADERSTART(1) RECEIVERSTART(6)
RECEIVERLENGTH(14);
TRAILERLITERAL(O*N95) TRAILERSTART(1);
LITERAL(O*N01) LITERALSTART(1) DROP(Y);
LITERAL(O*N99) LITERALSTART(1) DROP(Y);
```

Example 3: Sample JCL to send a compressed file with compression turned off in the TDClient step. See example 2, above, to compress a file prior to the TDClient step.

```
//STEP0020  EXEC PGM=EA2KMVSC,REGION=4M,TIME=1440,
//          PARM='CMDFILE=DD:CMDSEND'
//STEPLIB   DD DSN=SAIG.EASYACC.LOADLIB,DISP=SHR
//*
//EASYACC   DD DSN=SAIG.TDCLIENT.INI,DISP=SHR
//*
//EXFER     DD DSN=SAIG.EASYACC.EXFER.INI,DISP=SHR
//*
//CMDSEND   DD *
NETWORK=SAIGPORTAL FTPUSERID=TGxxxxx RESET
TRANSFER=(NAME=comp SENDUSERID=TGxxxxx SEND=DD:SENDFILE
COMPRESS=N)
//*
//SENDFILE   DD DSN=SAIG.D110501.FILTER,DISP=SHR
//*
//WORK04    DD DISP=NEW,UNIT=SYSDA,SPACE=(CYL,(5,5)),
//             LRECL=8192,BLKSIZE=0,RECFM=VB
//EASTATUS  DD DSN=SAIG.EASTATUS,
//             DISP=(NEW,DELETE),UNIT=SYSDA,SPACE=(CYL,(5,5)),
//             LRECL=8192,BLKSIZE=0,RECFM=VB,RETPD=60
//OUTMSG    DD SYSOUT=*
//SYSPRINT  DD SYSOUT=*
//EAFTPLOG  DD SYSOUT=*
//CPFTPLOG DD SYSOUT=*
```

Example 4: Sample JCL to pre-define files required by TDClient prior to sending.

```
//*****************************************************************
//*       Run IEBGENER to create your SYSUT1 file
*
//*****************************************************************
//SYSUT1   EXEC  PGM=IEBGENER
//SYSUT1   DD    DUMMY
//SYSUT2   DD    DSN=your.dataset.SYSUT1.file,
//               DISP=(NEW,CATLG),UNIT=SYSDA,
//               DCB=(LRECL=8192,BLKSIZE=0,RECFM=VB),
//               SPACE=(TRK,(5,5))
//SYSIN    DD    DUMMY
//*****************************************************************
//*       Run IEBGENER to create your SYSUT2 file
*
//*****************************************************************
//SYSUT2   EXEC  PGM=IEBGENER
//SYSUT1   DD    DUMMY
//SYSUT2   DD    DSN=your.dataset.SYSUT2.file,
//               DISP=(NEW,CATLG),UNIT=SYSDA,
//               DCB=(LRECL=8192,BLKSIZE=0,RECFM=VB),
//               SPACE=(TRK,(5,5))
//SYSIN    DD    DUMMY
//*****************************************************************
//*       Run IEBGENER to create your WORK01 file
*
//*****************************************************************
//WORK01   EXEC  PGM=IEBGENER
//SYSUT1   DD    DUMMY
//SYSUT2   DD    DSN=your.dataset.WORK01.file,
//               DISP=(NEW,CATLG),UNIT=SYSDA,
//               DCB=(LRECL=8192,BLKSIZE=0,RECFM=VB),
//               SPACE=(TRK,(5,5))
//SYSIN    DD    DUMMY
//*****************************************************************
//*       Run IEBGENER to create your WORK02 file
*
//*****************************************************************
//WORK02   EXEC  PGM=IEBGENER
//SYSUT1   DD    DUMMY
//SYSUT2   DD    DSN=your.dataset.WORK02.file,
//               DISP=(NEW,CATLG),UNIT=SYSDA,
//               DCB=(LRECL=8192,BLKSIZE=0,RECFM=VB),
//               SPACE=(TRK,(5,5))
//SYSIN    DD    DUMMY
//*****************************************************************
//*       Run IEBGENER to create your WORK03 file
*
//*****************************************************************
//WORK03   EXEC  PGM=IEBGENER
//SYSUT1   DD    DUMMY
```

```
//SYSUT2   DD    DSN=your.dataset.WORK03.file,
//              DISP=(NEW,CATLG),UNIT=SYSDA,
//              DCB=(LRECL=8192,BLKSIZE=0,RECFM=VB),
//              SPACE=(TRK,(5,5))
//SYSIN    DD    DUMMY
//*****************************************************************
//*       Run IEBGENER to create your WORK04 file
*
//*****************************************************************
//WORK04   EXEC  PGM=IEBGENER

//SYSUT1   DD    DUMMY
//SYSUT2   DD    DSN=your.dataset.WORK04.file, //
     DISP=(NEW,CATLG),UNIT=SYSDA,
//              DCB=(LRECL=8192,BLKSIZE=0,RECFM=VB),
//              SPACE=(TRK,(5,5))
//SYSIN    DD    DUMMY
//*****************************************************************
//*       Run IEBGENER to create your EASTATUS file
*
//*****************************************************************
//EASTATUS EXEC  PGM=IEBGENER
//SYSUT1   DD    DUMMY
//SYSUT2   DD    DSN=your.dataset.EASTATUS.file,
//              DISP=(NEW,CATLG),UNIT=SYSDA,
//              DCB=(LRECL=8192,BLKSIZE=0,RECFM=VB),
//              SPACE=(CYL,(5,5))
//SYSIN    DD    DUMMY
//*****************************************************************
//*       Run EA2KMVSC To Send
*
//*****************************************************************
//STEP0020       EXEC
     PGM=EA2KMVSC,REGION=4M,PARM='CMDFILE=DD:CMDSEND'
//*
//STEPLIB        DD    DSN=your.dataset.prefix.TDLOAD,DISP=SHR
//*
//EASYACC        DD    DSN=your.dataset.prefix.TDCLIENT.INI,DISP=SHR
//*
//EXFER          DD
     DSN=your.dataset.prefix.TDCLIENT.EXFER.INI,DISP=SHR
//*
//CMDSEND  DD    *
NETWORK=SAIGPORTAL FTPUSERID=TGxxxxx RESET
TRANSFER=(NAME=xxxxxxxx SEND=DD:SENDFLE
SENDUSERID=TGxxxxx OTHER_COMP_PARMS='SECFILE=DD:SECFILEX')
//*
//SENDFLE        DD    DSN=your.dataset.SEND.file,DISP=SHR
//SYSUT1    DD   DSN=your.dataset.SYSUT1.file,
//                  DISP=(OLD,DELETE,DELETE)          <=== CAN KEEP
//SYSUT2    DD   DSN=your.dataset.SYSUT2.file,
//                  DISP=(OLD,DELETE,DELETE)          <=== CAN KEEP
```

```
//WORK01    DD   DSN=your.dataset.WORK01.file,
//                  DISP=(OLD,DELETE,DELETE)          <=== CAN KEEP
//WORK02    DD   DSN=your.dataset.WORK02.file,
//                  DISP=(OLD,DELETE,DELETE)          <=== CAN KEEP
//WORK03    DD   DSN=your.dataset.WORK03.file,
//                  DISP=(OLD,DELETE,DELETE)          <=== CAN KEEP

//WORK04    DD   DSN=your.dataset.WORK04.file,
//                  DISP=(OLD,DELETE,DELETE)          <=== CAN KEEP
//EASTATUS  DD   DSN=your.dataset.EASTATUS.file,
//                  DISP=(OLD,DELETE,DELETE)          <=== CAN KEEP
//SECFILEX  DD *
SENDER(TGxxxxx);
HEADERLITERAL(O*N05) HEADERSTART(1) RECEIVERSTART(6)
RECEIVERLENGTH(14)
CLASSSTART(25) CLASSLENGTH(8);
TRAILERLITERAL(O*N95) TRAILERSTART(1);
LITERAL(O*N01) LITERALSTART(1) DROP(Y);
LITERAL(O*N99) LITERALSTART(1) DROP(Y);
//*
//COMPLOG   DD SYSOUT=*
//OUTMSG    DD SYSOUT=*
//SYSPRINT  DD SYSOUT=*
//EAFTPLOG  DD SYSOUT=*
//EALOG     DD SYSOUT=*
//EXFERLOG  DD SYSOUT=*
//CPFTPLOG  DD SYSOUT=*
//*
```

Appendix B – Command Lines for Different Methods of Receiving Data

Since FTPPASSWD is stored in the TDClient.INI file in encrypted format, when you change your password, it is not necessary to hardcode your password in all of your scripts and/or JCL. See Chapter 3 for recommendation to change your password.

Note: These are examples only and may require customization at your site.

Example 1: Receive all data by specific sender ID: RECEIVEUSERID= the sender of the data being requested.

```
NETWORK=SAIGPORTAL FTPUSERID=TGxxxxx RESET
TRANSFER=(NAME=name RECEIVE=name RECEIVEUSERID=TGxxxx)
```

Example 2: Receive all data by specific message class:

RECEIVECLASS= the message class of the data being requested.

```
NETWORK=SAIGPORTAL FTPUSERID=TGxxxxx RESET
TRANSFER=(NAME=name RECEIVE=name RECEIVECLASS=messclass)
```

Example 3: Receive all data in mailbox: Notice that RECEIVEUSERID= and RECEIVECLASS= have been removed.

```
NETWORK=SAIGPORTAL FTPUSERID=TGxxxxx RESET
TRANSFER=(NAME=name RECEIVE=name)
```

Example 4: Receive data by unique file name: RECEIVE_SERVER_FILE= the Unique Filename on TDCM or the Available status record within your Query List. (See **Figure 35**)

```
NETWORK=SAIGPORTAL FTPUSERID=TGxxxxx RESET
TRANSFER=(NAME=name RECEIVE=name
RECEIVE_SERVER_FILE=xxxxxxxxxxxxxxxxxxx)
```

Example 5: Delete data by unique file name: DELETE_SERVER_FILE= the Unique Filename on TDCM or the Available status record within your Query List. (See **Figure 35**)

```
NETWORK=SAIGPORTAL FTPUSERID=TGxxxxx RESET
TRANSFER=(NAME=name RECEIVE=name
     DELETE_SERVER_FILE=xxxxxxxxxxxxxxxxxxxx
```

Example 6: Receive multiple files by specific message class using multiple TRANSFER statements. Specify RECEIVECLASS= for the message class of the data being requested. You must specify a corresponding output file for each Transfer statement. We recommend that you specify the exact record length if receiving fixed block data.

```
NETWORK=SAIGPORTAL FTPUSERID=TGxxxxx RESET
TRANSFER=(NAME=nam1 RECEIVE=RECVFL1 RECEIVECLASS=SARA02OP)
TRANSFER=(NAME=nam2 RECEIVE=RECVFL2 RECEIVECLASS=CORR02OP)
```

Example 7: Concatenate the O*N01 File Header and O*N99 File Trailer records into your Receive data file using IEBGENER. Sample for users whose programs require the O*N01 & O*N99 records.

```
//STEP01    EXEC PGM=IEBGENER
//SYSPRINT  DD SYSOUT=*
//SYSUT1    DD DSN=your.dataset.prefix.IEBGEN01,DISP=SHR
//SYSUT2    DD DSN=your.dataset.prefix.IEBGALL.RECV1,
//             DISP=(MOD,CATLG),UNIT=SYSDA,
//             DCB=(LRECL=nnnn,BLKSIZE=nnnnn,RECFM=FB),
//             SPACE=(CYL,(n,n))
//SYSIN     DD DUMMY
//*
//STEP02 EXEC PGM=EA2KMVSC,REGION=4M,PARM='CMDFILE=DD:CMDRECV'
//*
//STEPLIB   DD DSN=your.dataset.prefix.TDLOAD,DISP=SHR
//*
//EASYACC   DD DSN=your.dataset.prefix.TDCLIENT.INI,DISP=SHR
//*
//EXFER     DD DSN=your.dataset.prefix.TDCLIENT.EXFER.INI,DISP=SH
//*
//CMDRECV DD *
NETWORK=SAIGPORTAL FTPUSERID=TGxxxxx RESET
TRANSFER=(NAME=name RECEIVE=DD:receive RECEIVEUSERID=TGxxxxx)
//*
//receive   DD DSN=your.dataset.prefix.IEBGALL.RECV1,
//             DISP=(MOD,CATLG),UNIT=SYSDA,
//             DCB=(LRECL=nnnn,BLKSIZE=nnnnn,RECFM=FB),
//             SPACE=(CYL,(nn,nn))
//*
//SYSUT1    DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//             LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK01    DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//             LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK02    DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//             LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK03    DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//             LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK04    DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//             LRECL=8192,BLKSIZE=0,RECFM=VB
//EASTATUS  DD DSN=your.dataset.prefix.EASTATUS,
//             DISP=(NEW,CATLG),UNIT=SYSDA,SPACE=(CYL,(5,5)),
//             LRECL=8192,BLKSIZE=0,RECFM=VB
 //*
 //DCMPLOG   DD SYSOUT=*
 //OUTMSG    DD SYSOUT=*
 //SYSPRINT  DD SYSOUT=*
 //EAFTPLOG  DD SYSOUT=*
 //EALOG     DD SYSOUT=*
 //EXFERLOG  DD SYSOUT=*
```

```
//STEP03 EXEC PGM=IEBGENER,COND=(00,NE,STEP02)
//SYSPRINT  DD SYSOUT=*
//SYSUT1    DD
DSN=your.dataset.prefix.IEBGALL.RECV1,DISP=(OLD,PASS)
//          DD DSN=your.dataset.prefix.IEBGEN99,DISP=SHR
//SYSUT2    DD DSN=your.dataset.prefix.IEBGNOC.RECV6,
//             DISP=(,CATLG),UNIT=SYSDA,
//             DCB=(your.dataset.prefix.IEBGALL.RECV1),
//             SPACE=(CYL,(nn,nn))
//SYSIN     DD DUMMY
//*
//STEP04 EXEC PGM=IEFBR14,COND=(00,NE,STEP03)
//FILE1     DD DSN=your.dataset.prefix.IEBGALL.RECV1,
//             DISP=(MOD,DELETE),
//             SPACE=(CYL,(nn,nn)
```

Example 8: Sample JCL to receive a file with decompression turned off. See example 9 to decompress file later.

Note: These are examples only and may require customization at your site.

```
//STEP0020 EXEC
PGM=EA2KMVSC,REGION=4M,PARM='CMDFILE=DD:CMDRECV'
//*
//STEPLIB DD DSN=SAIG.EASYACC.LOADLIB,DISP=SHR
//EASYACC DD DSN=SAIG.D081501.APPSYS.TDCLIENT.INI,DISP=SHR
//EXFER DD DSN=SAIG.TDCLIENT.EXFER.INI,DISP=SHR
//*************************************************************
//* THESE COMMANDS WILL RECEIVE MULTIPLE FILES AND PUT INTO
THE
//* RECEIVE FILE. BUT, WHEN YOU DECOMPRESS THE FILE YOU MUST
//* HAVE THE EXACT DECOMPRESS LRECL OR THE O*N05 WILL NOT
BEGIN IN
//* POSITION 1. IT WILL INSTEAD BE AT THE END OF O*N95.
//*************************************************************
//CMDRECV DD *
NETWORK=SAIGPORTAL FTPUSERID=TGxxxxx RESET
TRANSFER=(NAME=COPY RECEIVE=DD:RECV
OTHER_DECOMP_PARMS='COPYONLY')
//*
//RECV DD DSN=SAIG.COMP.CORR03OP,
//     DISP=(NEW,CATLG),UNIT=SYSDA,SPACE=(TRK,(5,5)),
//     LRECL=8192,BLKSIZE=0,RECFM=VB
//*
//SYSUT1 DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//       LRECL=8192,BLKSIZE=0,RECFM=VB
//SYSUT2 DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//       LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK01 DD DSN=SAIG.WORK01.D112701.ALL,
//       DISP=(NEW,CATLG),UNIT=SYSDA,SPACE=(TRK,(5,5)),
//       LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK02 DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
```

```
//        LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK03 DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//        LRECL=8192,BLKSIZE=0,RECFM=VB
//WORK04 DD DISP=NEW,UNIT=SYSDA,SPACE=(TRK,(5,5)),
//        LRECL=8192,BLKSIZE=0,RECFM=VB
//EASTATUS DD SYSOUT=*
//OUTMSG DD SYSOUT=*
//DCMPLOG DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//EAFTPLOG DD SYSOUT=*
//
```

Example 9: Sample JCL to decompress a file that has already been received by TDClient with decompression turned off. When you decompress the file you must have the exact decompressed record length of the file or the O*N05 record will not begin in position one. It will instead be at the end of the O*N95 record.

Note: These are examples only and may require customization at your site.

```
//STEP0010  EXEC PGM=DECOMP,REGION=4M,TIME=1440,
//          PARM='APPEND UNCOMP'
//STEPLIB   DD DSN=SAIG.EASYACC.LOADLIB,DISP=SHR
//*
//SYSOUT   DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//DCMPLOG  DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//DATAIN      DD DSN=SAIG.COMP.CORR03OP,DISP=SHR
//DATAOT      DD DSN=SAIG.UNCOMP.CORR03OP.DCMP3,
//            DISP=(NEW,CATLG),RETPD=30,
//            SPACE=(CYL,(60,60)),UNIT=SYSDA,
//            LRECL=2850,BLKSIZE=0,RECFM=FB
```

Example 10: Sample JCL to pre-define files required by TDClient prior to receiving.

```
//*************************************************************
//*       Run IEBGENER to create your RECVFLE file
//*************************************************************
//RECVFLE    EXEC  PGM=IEBGENER
//SYSUT1     DD    DUMMY
//SYSUT2     DD    DSN=your.dataset.receive.file,
//                 DISP=(NEW,CATLG),UNIT=SYSDA,
//                 DCB=(LRECL=nnnn,BLKSIZE=nnnnn,RECFM=FB),
//                 SPACE=(CYL,(nn,nn))
//SYSIN      DD    DUMMY
//*************************************************************
//*       Run IEBGENER to create your SYSUT1 file
//*************************************************************
//SYSUT1     EXEC  PGM=IEBGENER
//SYSUT1     DD    DUMMY
//SYSUT2     DD    DSN=your.dataset.SYSUT1.file,
//                 DISP=(NEW,CATLG),UNIT=SYSDA,
//                 DCB=(LRECL=8192,BLKSIZE=0,RECFM=VB),
//                 SPACE=(TRK,(5,5))
//SYSIN      DD    DUMMY
//*************************************************************
//*       Run IEBGENER to create your SYSUT2 file
//*************************************************************
//SYSUT2     EXEC  PGM=IEBGENER
//SYSUT1     DD    DUMMY
//SYSUT2     DD    DSN=your.dataset.SYSUT2.file,
//                 DISP=(NEW,CATLG),UNIT=SYSDA,
//                 DCB=(LRECL=8192,BLKSIZE=0,RECFM=VB),
//                 SPACE=(TRK,(5,5))
//SYSIN      DD    DUMMY
//*************************************************************
//*       Run IEBGENER to create your WORK01 file
//*************************************************************
//WORK01     EXEC  PGM=IEBGENER
//SYSUT1     DD    DUMMY
//SYSUT2     DD    DSN=your.dataset.WORK01.file,
//                 DISP=(NEW,CATLG),UNIT=SYSDA,
//                 DCB=(LRECL=8192,BLKSIZE=0,RECFM=VB),
//                 SPACE=(TRK,(5,5))
//SYSIN      DD    DUMMY
//*************************************************************
//*       Run IEBGENER to create your WORK02 file
//*************************************************************
//WORK02     EXEC  PGM=IEBGENER
//SYSUT1     DD    DUMMY
//SYSUT2     DD    DSN=your.dataset.WORK02.file,
//                 DISP=(NEW,CATLG),UNIT=SYSDA,
//                 DCB=(LRECL=8192,BLKSIZE=0,RECFM=VB),
//                 SPACE=(TRK,(5,5))
//SYSIN      DD    DUMMY
//*************************************************************
//*       Run IEBGENER to create your WORK03 file
```

```
//*************************************************************
//WORK03     EXEC  PGM=IEBGENER
//SYSUT1     DD    DUMMY
//SYSUT2     DD    DSN=your.dataset.WORK03.file,
//                 DISP=(NEW,CATLG),UNIT=SYSDA,
//                 DCB=(LRECL=8192,BLKSIZE=0,RECFM=VB), //
 SPACE=(TRK,(5,5))
//SYSIN      DD    DUMMY
//*************************************************************
//*       Run IEBGENER to create your WORK04 file
//*************************************************************
//WORK04     EXEC  PGM=IEBGENER
//SYSUT1     DD    DUMMY
//SYSUT2     DD    DSN=your.dataset.WORK04.file,
//                 DISP=(NEW,CATLG),UNIT=SYSDA,
//                 DCB=(LRECL=8192,BLKSIZE=0,RECFM=VB),
//                 SPACE=(TRK,(5,5))
//SYSIN      DD    DUMMY
//*************************************************************
//*       Run IEBGENER to create your EASTATUS file
//*************************************************************
//EASTATUS   EXEC  PGM=IEBGENER
//SYSUT1     DD    DUMMY
//SYSUT2     DD    DSN=your.dataset.EASTATUS.file,
//                 DISP=(NEW,CATLG),UNIT=SYSDA,
//                 DCB=(LRECL=8192,BLKSIZE=0,RECFM=VB),
//                 SPACE=(CYL,(5,5))
//SYSIN
DD     DUMMY
//*************************************************************
//*       Run EA2KMVSC To Receive
//*************************************************************
//STEP0020   EXEC
 PGM=EA2KMVSC,REGION=4M,PARM='CMDFILE=DD:CMDRECV'
//*
//STEPLIB    DD    DSN=your.dataset.prefix.TDLOAD,DISP=SHR
//*
//EASYACC    DD    DSN=your.dataset.prefix.TDCLIENT.INI,DISP=SHR
//*
//EXFER      DD
 DSN=your.dataset.prefix.TDCLIENT.EXFER.INI,DISP=SHR
//*
//CMDRECV    DD    *
NETWORK=SAIGPORTAL FTPUSERID=TGxxxxx RESET
TRANSFER=(NAME=xxxxxxxx RECEIVE=DD:RECVFLE
RECEIVEUSERID=TGxxxxx RECEIVECLASS=messageclass)
//*
//RECVFLE    DD    DSN=your.dataset.receive.file,DISP=SHR
//*
//SYSUT1     DD    DSN=your.dataset.SYSUT1.file,
//                   DISP=(OLD,DELETE,DELETE)          <=== CAN
KEEP
//SYSUT2     DD    DSN=your.dataset.SYSUT2.file,
//                   DISP=(OLD,DELETE,DELETE)          <=== CAN
KEEP
```

```
//WORK01     DD     DSN=your.dataset.WORK01.file,
//                    DISP=(OLD,DELETE,DELETE)            <=== CAN
KEEP
//WORK02     DD     DSN=your.dataset.WORK02.file,
//                    DISP=(OLD,DELETE,DELETE)            <=== CAN
KEEP
//WORK03     DD     DSN=your.dataset.WORK03.file,
//                    DISP=(OLD,DELETE,DELETE)            <=== CAN
KEEP
//WORK04     DD     DSN=your.dataset.WORK04.file,
//                    DISP=(OLD,DELETE,DELETE)            <=== CAN
KEEP
//EASTATUS   DD     DSN=your.dataset.EASTATUS.file,
//                    DISP=(OLD,DELETE,DELETE)            <=== CAN
KEEP
//DCMPLOG    DD SYSOUT=*
//OUTMSG     DD SYSOUT=*
//SYSPRINT   DD SYSOUT=*
//EAFTPLOG   DD SYSOUT=*
//EALOG      DD SYSOUT=*
//EXFERLOG   DD SYSOUT=*
//*
```

Appendix C – UNIX/LINUX Scripts for Different Methods of Receiving Data

Note: These are examples only and may require customization at your site.

Example 1: Receive data by a specific sender ID, RECEIVEUSERID= the sender of the data being requested. Depending on options in the TDClient.INI file it will either all be concatenated into one file (if APPEND=Y and AUTOEXT=N) or each item will go into separate files named receive.txt.xxxx, where xxxx can be any number from 0001 to 9999 (if APPEND=N and AUTOEXT=4.)

```
tdclientc network=saigportal ftpuserid=TGxxxxx reset\
"transfer=(name=xxxxx
receive=./path/to/the/file/to/receive.txt
\receiveuserid=TGxxxxx)"
```

Example 2: Receive all data by a specific message class, RECEIVECLASS= the message class of the data being requested. Depending on options in the TDClient.INI file it will either all be concatenated into one file (if APPEND=Y and AUTOEXT=N) or each item will go into separate files named receive.txt.xxxx, where xxxx can be any number from 0001 to 9999 (if APPEND=N and AUTOEXT=4.)

```
tdclientc network=saigportal ftpuserid=TGxxxxx reset\
"transfer=(name=xxxxxxxx
receive=./path/to/the/file/to/receive.txt
\receiveclass=xxxxxxxx)"
```

Example 3: Receive data by unique file name: RECEIVE_SERVER_FILE= the Unique Filename on TDCM or the Available status record within your Query List.

```
tdclientc network=saigportal ftpuserid=TGxxxxx reset\
"transfer=(name=xxxxxx
receive=./path/to/the/file/to/receive.txt
\receive_server_file= xxxxxxxxxxxxxxxxxxx)"
```

Example 4: Receive all data in mailbox: Notice that RECEIVEUSERID= and RECEIVECLASS= have been removed.

```
tdclientc network=saigportal ftpuserid=TGxxxxx reset\
"transfer=(name=xxxxxxxx\
receive=./path/to/the/file/to/receive.txt)"
```

Appendix D – Windows Command Lines for Different Methods of Receiving Data

Note: These are examples only and may require customization at your site.

Example 1: Receive data by a specific sender ID, RECEIVEUSERID= the sender of the data being requested. Depending on options in the TDClient.INI file it will either all be concatenated into one file (if APPEND=Y and AUTOEXT=N) or each item will go into separate files named receive.txt.xxxx, where xxxx can be any number from 0001 to 9999 (if APPEND=N and AUTOEXT=4.)

```
tdclientc network=saigportal ftpuserid=TGxxxxx reset
"transfer=(name=xxxxx
receive=.\path\to\the\file\to\receive.txt
receiveuserid=TGxxxxx)"
```

Example 2: Receive all data by a specific message class, RECEIVECLASS= the message class of the data being requested. Depending on options in the TDClient.INI file it will either all be concatenated into one file (if APPEND=Y and AUTOEXT=N) or each item will go into separate files named receive.txt.xxxx, where xxxx can be any number from 0001 to 9999 (if APPEND=N and AUTOEXT=4.)

```
tdclientc network=saigportal ftpuserid=TGxxxxx reset\
"transfer=(name=xxxxxxxx
receive=.\path\to\the\file\to\receive.txt
receiveclass=xxxxxxxx)"
```

Example 3: Receive data by unique file name: RECEIVE_SERVER_FILE= the Unique Filename on TDCM or the Available status record within your Query List.

```
tdclientc network=saigportal ftpuserid=TGxxxxx reset
"transfer=(name=xxxxxx
receive=.\path\to\the\file\to\receive.txt
\receive_server_file= xxxxxxxxxxxxxxxxxxx)"
```

Example 4: Receive all data in mailbox: Notice that RECEIVEUSERID= and RECEIVECLASS= have been removed.

```
tdclientc network=saigportal ftpuserid=TGxxxxx reset
"transfer=(name=xxxxxxxx
   receive=.\path\to\the\file\to\receive.txt)"
```

Appendix E – Troubleshooting

Listed below are common errors received when sending and receiving data. We are providing you with common resolutions to these errors. Users can view the SYSOUT file for details of return codes and ftp errors received. Midrange users can view the Temp directory for files with return codes.

Table 12: Errors Received When Sending and Receiving Data

Error	Resolution
RC=03	Can indicate an invalid command line argument. This error is often received because of missing parameters or incorrect syntax in the command line. A missing "SENDUSERID" can generate this error. For instance: NETWORK=SAIGPORTAL FTPUSERID=TG50000 FTPPASSWD=PASSWORD RESET TRANSFER=(NAME=anything SEND=sendfilename **SENDUSERID=TG50000** OTHER_COMP_PARMS='SECFILE=secfilename')
RC=05	Can indicate a missing network header (O*N05) or trailer (O*N95) in the data file; or the SECFILE may have a syntax error. May indicate and invalid password.
RC=32 Error writing output file WORK01 WARNING: Compression of file failed Error: Compress of file failed Compress failed. Error Code: CMP0033	Resolution is to increase space for WORK01 or receive data by tape.
RC=103	Decompression error; check the SECFILE for syntax errors.
421 Peer Closed Connection	If you receive the error "421 peer closed connection" when receiving files then your internet connection has dropped. Connect again and you should receive as normal. To prevent manually restarting your job in the future, make sure your TDCLIENT.INI file has the following parameters specified: AUTO_RETRY=Y MAX_RETRY=5 **RETRY_DELAY=30**
425	Make sure these parameters are in the Tdclient.ini file, and are set to "Y": **COMMAND_OVER_DATA=Y** **DATA_OVER_COMMAND=Y**
332 or 531 Change password required	Your SAIG password has expired. Your must change your mailbox password every 90 days.
532 through 537 or 540 Login incorrect	You are using an incorrect SAIG password. (See the EDconnect Error Code List for a complete description of each error.)

Error	Resolution
550 Internal Server Problem Login for UserID failed	A lower case “tg” was used for the user ID. Use “TG” in upper case for the user ID.
Autoextent function not working properly	When setting options in the TDClient.INI file to make separate files for every file received(AUTOEXT=Y APPEND=N), the files aren’t being created correctly. The 05 header is being placed in one file, and the data and 95 trailer is getting placed in a different file. Solution is to set AUTOEXT=N APPEND=Y to associated file, and receive multiple files. Designate file name to append all files plus N05.
B37 U4083; SOC4; CEE3250C The system or user abend SB37 R=00000004 was issued. From entry point DCCloseOutputFile at compile unit offset.	We have experienced multiple reasons for this error. Solutions may include writing to Tape, removing the RLSE command (with VAN reference), or pre-defining datasets prior to EA step. You can try one of the following solutions: If using the RLSE parameter in your in DD definition, i.e., SPACE=(CYL,(nn,nn),RLSE), you may receive SB37 errors when receiving multiple batches. TDClient opens the Receive file for the first batch to be received and then closes the file releasing unused space. When the next batch is received EA opens the Receive file again with a disposition of MOD, which appends the new batch to the data already in the file. Since the remaining space allocated was released after the first batch was received this may cause a space problem. A similar situation may occur with the additional files required by EA for a Send or a Receive, SYSUT1, SYSUT2, WORK01, WORK02, WORK03, WORK04 and EASTATUS. We recommend that you pre-define these datasets prior to the TDClient step. You can then use a MOD disposition to append multiple batches to the file. See Appendix B, Example 10 for a sample using IEBGENER to pre-define datasets. There is a known issue with EA resulting in an abend code U4083 or SOC4. This is caused by a storage overlay. Click Commerce has repaired the problem in version 1.5 of EA. A temporary solution is to pre-define the files used by TDClient. See solution number 1 above.
CEE3512S-message (IBM LE error)	Caused by not having valid USS home directories setup for the user IDs. According to IBM, the system was searching the USS files when trying to load COMMPRSS, but was failing when searching the user’s home directory. In later releases of OS/390 the search continues to MVS locations such as steplib, Ipa, and linklist, but 2.6 causes an abend.
EDC81281 Connection Refused	In MVS 2.8 LE 1.9+, an LE(Language Environment) error. IT staff need to look at the LE setup.
<u>MVS v2.6 Only</u> Error Code: DCM0061 WARNING: Decompression of file failed Error: Decompression of file failed Decomp failed. Error	Add DSN=&&WORK01(etc) to the WORK01 DD, WORKxx, and SYSUTx DDs in the RECV job.

Error	Resolution
Code: DCM0061 WARNING: Decompression of file failed Made backup copy of compressed file which failed to decompress. Backup FileName: WORK01.001	
RC=U4038	This is usually an OMVS security error. Resubmitting the job is successful. Click Commerce recommends following the installation and customization instructions in the OS/390 VnRn.n UNIX System Services Planning manual, which can be found in the IBM documentation CDs (usually disk 1) in the OS/390 VnRn.n UNIX System Services Base Element Bookshelf. Be sure to use the manual that corresponds to your operating system release. If your site is non-UNIX, and you don't want to set up OMVS for your users, B-trade recommends setting up a default OMVS segment for all users, so batch programs that make use of OMVS Services and Functions can be run.
RC=U4038	This is usually an OMVS security error. Resubmitting the job is successful. Click Commerce recommends following the installation and customization instructions in the OS/390 VnRn.n UNIX System Services Planning manual, which can be found in the IBM documentation CDs (usually disk 1) in the OS/390 VnRn.n UNIX System Services Base Element Bookshelf. Be sure to use the manual that corresponds to your operating system release. If your site is non-UNIX, and you don't want to set up OMVS for your users, B-trade recommends setting up a default OMVS segment for all users, so batch programs that make use of OMVS Services and Functions can be run.
UNIX/LINUX only Id.so.1: ea2ksunc: fatal: libcpsql.so: open failed: No such file or directory Chngpasswd: 18540 Killed	The solution is to copy the shared objects to the shared library. Then, issue a chmod on the files from the lib directory(chmod –R 755 libcpsql.so). **Note:** the lib directory can be located from the root (/lib).libcpsql.so, if this object is not located in the shared library then please contact customer service.
UNIX/LINUX only Decomp failed, Error	Your directory does not exist. Ensure that the AUTOEXT parameter in the TDCLIENT.INI file is set to

Error	Resolution
Code: DCM0073	N.(AUTOEXT=N) and the APPEND is set to Y (APPEND=Y). In your Transfer statement, make sure the RECEIVE= parm has a directory name, not a filename specified (receive=./incoming/sarafiles/).
WARNING: Action failed Unable to RESET TDClient Restart file Failed to open EA Restart file [./eastatus.txt] in write mode permission denied\	Solution is to create the eastatus.txt in the TDCLIENT.INI directory, with read & write permissions.
Windows only Executes prior transmission when sending or receiving a new file.	Slashes (/) used in the command line are backwards and should be facing the opposite direction (\). The system will not execute the line and submits the last transmission that was stored in the TDClient.ini file. Check the Response.log file, located in the directory where the TDClient has been installed, for errors.

Appendix F – TDClient.ini Defaults

```
[SECURITY]
NETWORK=SAIGPORTAL
EDINAME=SAIG
EMAILADDRESS=CPSSAIG@ED.GOV
RTMGENERATE=N
AUTOUPDATERUNTIME=N
MODULUS=0
APPROVALCODE=
EXPDATE=
VALID=FULL VERSION

[NETWORKS]
1=SAIGPORTAL
2=SAIGPORTALDEV

[MAINT]
NETWORK=SAIGPORTAL

[SAIGPORTAL]
HOSTIPNAME=SAIGMAILBOX.ED.GOV
NETWORKSTYLE=EAFTP
PASSIVE=N
CASE=
DHONLY=Y
SSL=Y
AUTO_DIAL=N
AUTO_DISCONNECT=N
SECURITYMENU=Y
SUNIQUE=0
CONTROL_PORT=26581
MAX_AUTO_DIAL_DELAY=180
USERID=
PASSWORD=
NAME=
AUTO_DELETE=N
COMMAND_OVER_DATA=Y
DATA_OVER_COMMAND=Y
SITEDELAY=0
ADDRESS_BOOK=N
PRIMARY_DIALER_TYPE=N
PRIMARY_DIALER_APP=
PRIMARY_DIAL_ENTRY=
SECONDARY_DIALER_TYPE=N
SECONDARY_DIALER_APP=
```

```
SECONDARY_DIAL_ENTRY=
MAILBOX_USERID=
MAILBOX_PASSWORD=
ACCOUNT=
HOSTIPNAME2=
GATEWAY_MAILBOX=
PROXY_TYPE=
PROXY_USERID=
PROXY_PASSWORD=
LOW_CLIENT_PORT=0
HIGH_CLIENT_PORT=0
TIMEOUT=0
USEDEFAULTSENDER=MISSING
USEDEFAULTAS2NAME=MISSING
SERVER_RESPONSE_BUFFERING=N
ENABLE_ALIAS_PROBE=N
USEDEFAULTAS2NAME=MISSING
USE_SMTP_LOGIN=N
SMTP_USERID=
SMTP_PASSWORD=

[SAIGPORTAL-DEFAULT_SENDPARMS]
COMPRESS=Y
FILTER=Y
ASCII=Y
CRLF=Y

[SAIGPORTAL-DEFAULT_RECEIVEPARMS]
APPEND=Y
AUTOEXT=N
UNCOMP=Y
ASCII=Y

[SAIGPORTALDEV]
HOSTIPNAME=SAIGMAILBOXDEV.ED.GOV
NETWORKSTYLE=EAFTP
PASSIVE=N
CASE=
DHONLY=Y
SSL=Y
AUTO_DIAL=N
AUTO_DISCONNECT=N
SECURITYMENU=Y
SUNIQUE=0
CONTROL_PORT=26581
MAX_AUTO_DIAL_DELAY=180
USERID=
```

```
PASSWORD=
NAME=
AUTO_DELETE=N
COMMAND_OVER_DATA=Y
DATA_OVER_COMMAND=Y
SITEDELAY=0
ADDRESS_BOOK=N
PRIMARY_DIALER_TYPE=N
PRIMARY_DIALER_APP=
PRIMARY_DIAL_ENTRY=
SECONDARY_DIALER_TYPE=N
SECONDARY_DIALER_APP=
SECONDARY_DIAL_ENTRY=
MAILBOX_USERID=
MAILBOX_PASSWORD=
ACCOUNT=
HOSTIPNAME2=
GATEWAY_MAILBOX=
PROXY_TYPE=
PROXY_USERID=
PROXY_PASSWORD=
LOW_CLIENT_PORT=0
HIGH_CLIENT_PORT=0
TIMEOUT=0
USEDEFAULTSENDER=MISSING
USEDEFAULTAS2NAME=MISSING
SERVER_RESPONSE_BUFFERING=N
ENABLE_ALIAS_PROBE=N
USEDEFAULTAS2NAME=MISSING
USE_SMTP_LOGIN=N
SMTP_USERID=
SMTP_PASSWORD=

[SAIGPORTALDEV-DEFAULT_SENDPARMS]
COMPRESS=Y
FILTER=Y
ASCII=Y
CRLF=Y

[SAIGPORTALDEV-DEFAULT_RECEIVEPARMS]
APPEND=Y
AUTOEXT=N
UNCOMP=Y
ASCII=Y

[IDENTIFY]
NETWORK=SAIGPORTAL
```

```
MULTITHREADED=N
DISABLE_DIALER=N
MULTIFILE=Y
AUTO_RETRY=Y
MAX_RETRY=5
RETRY_DELAY=300
AUDIT_START_DATE=
AUDIT_END_DATE=
STARTTIME=
STARTDATE=
LOG_MEM=N
LOG_INI=N
LOG_XFER=N
LOG_FTP=4
LOG_EASYACC=N
LOG_THREAD=N
MAX_AUTO_DIAL_DELAY=120
CONTINUE_AFTER_MAX_RETRY=Y
AUDIT_SENDER=
AUDIT_RECEIVER=
HELPFILENAME=tdclient.hlp
LOG_SOCKET=N
LOG_SMIME=N
WORK_RECORD_LENGTH=

[EAPATH]
BASEPATH=
```